Decision Trees and Industrial Machine Learning

Jeremy Barnes, Recoset
(jeremy@recoset.com)

November 9, 2010

www.recoset.com

Me

- BEng/BSc (1995-1999)
- Founded two companies:
  - Idilia (Montreal): Computational Linguistics (2000-2009)
  - Recoset (Montreal): Recommendations & Online Advertising (2009-)
- 10 years industrial machine learning

Contents

- Decision Trees
- Industrial Machine Learning
- My toolbox
- Applications: Computational Linguistics, Recommendations
- 4 stories: Tanks, Fraud, Terrorists, Monkeys

Story: Tanks in the Desert

Story: Tanks in the Desert

- Using ML requires insight

Story: Tanks in the Desert

- Using ML requires insight
- An algorithm is only a good as its data

Decision Trees

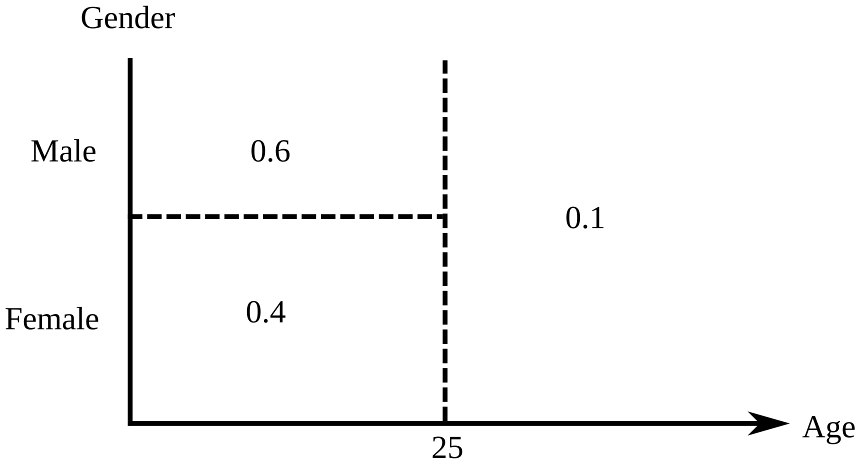
- A very simple Machine Learning algorithm
- Learns a hierarchical set of rules:

  if *predicate*
  then *action1*
  else *action2*

- for example

  if *gender=Male*
  then
      if *age < 25*
      then $p(crash) = 0.6$
      else $p(crash) = 0.1$
  else $p(crash) = 0.4$

Space Partitioning



How?

Greedy Algorithm to Separate Data

1. Learns the "best" split at the top level
   - Try every value of every variable
   - Choose the one that minimizes a cost function
2. Partition the dataset along the best split
3. Repeat (recursion) on each of the halves of the dataset
4. Stop when:
   - Maximum depth reached
   - Data is perfectly separated

Cost Functions

Regression

- Calculate label mean of each half of the data
- cost $=$ least squares between data and partition means

Classification

- Measure the impurity of each of the classes
- Gini, entropy, ...

Advantages

- Simple
- Fast
- Easy to understand output
- Can learn highly non-linear surfaces
- Works well with lots of dimensions (features)
- Not affected by redundant features

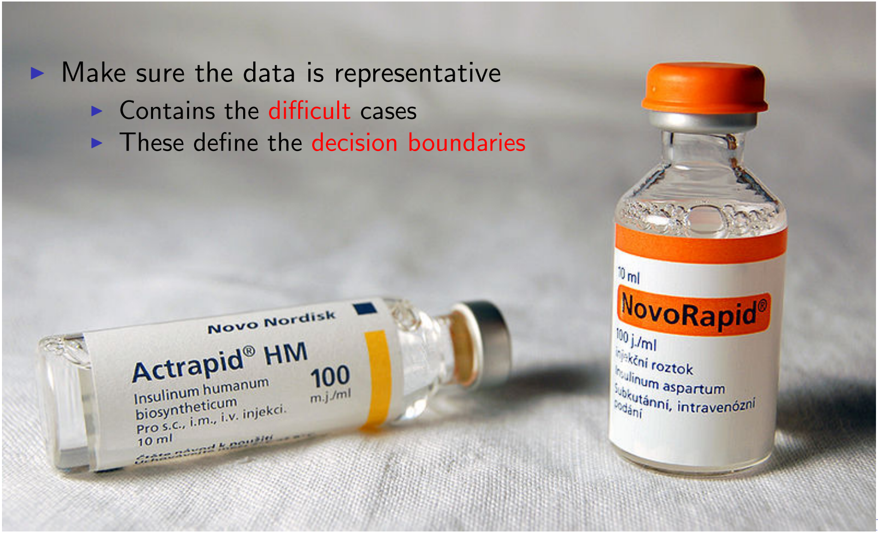
Disadvantages

- Sensitive to noise
- Cannot learn smooth functions
- Cannot extrapolate
- Requires enormous amounts of data for complex cases
- Decisions must be parallel to an axis

Story: Medical Expense Fraud

Story: Medical Expense Fraud

- Make sure the data is representative
  - Contains the difficult cases
  - These define the decision boundaries

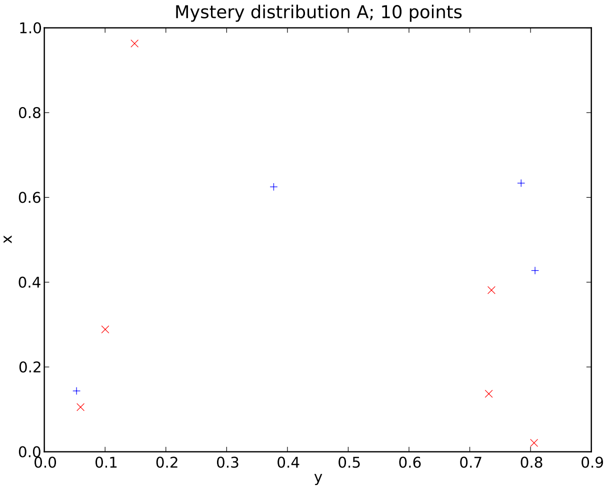


Story: Medical Expense Fraud

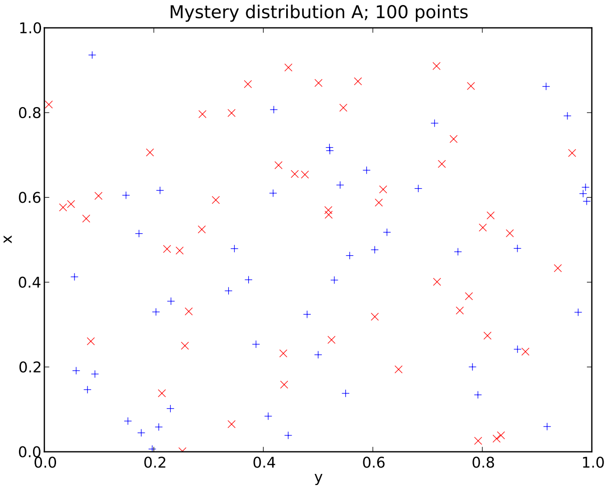
- Make sure the data is representative
  - Contains the difficult cases
  - These define the decision boundaries
- The right feature turns failure into success

Feature Engineering

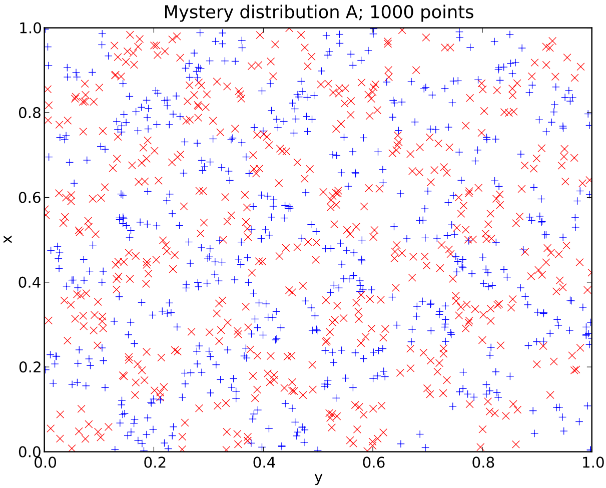
- Adding information to the dataset to make the classifier's job easier
- For example, adding "buys insulin regularly"
- Feature types
  - Derived from existing features
  - Added information
  - Art

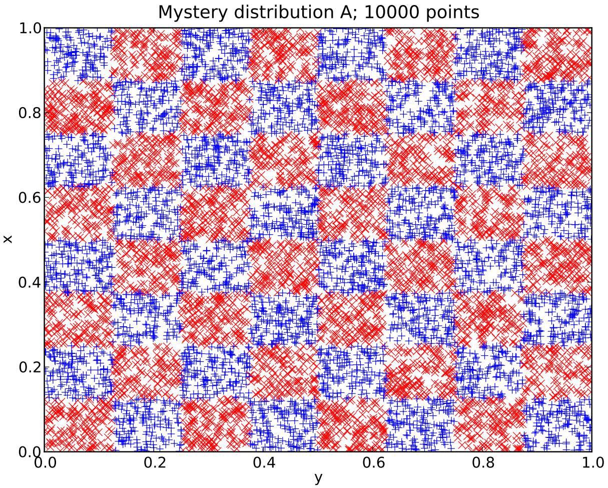


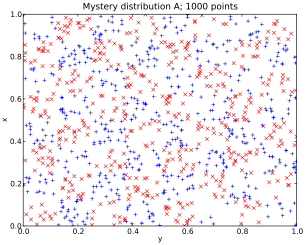
Mystery distribution A; 10 points
1.0
0.8
0.6
0.4
0.2
0.0
x
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
y



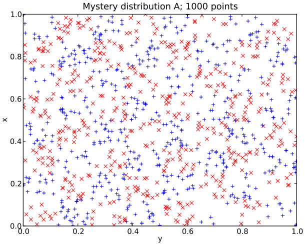
Mystery distribution A; 100 points
x
y
0.0
0.2
0.4
0.6
0.8
1.0


Mystery distribution A; 1000 points
x
y
0.0
0.2
0.4
0.6
0.8
1.0

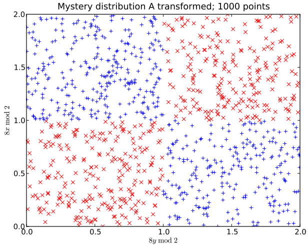

Mystery distribution A; 10000 points
1.0
0.8
0.6
0.4
0.2
0.0
x
0.0
0.2
0.4
0.6
0.8
1.0
y



Mystery distribution A; 1000 points
1.0
0.8
0.6
0.4
0.2
0.0
x
0.0
0.2
0.4
0.6
0.8
1.0
y



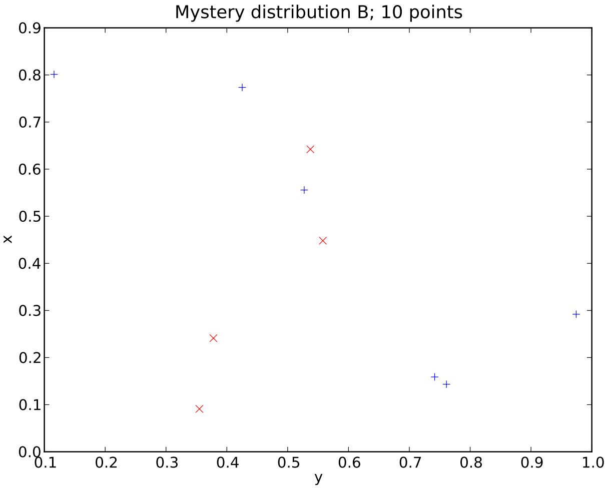
$\Rightarrow$

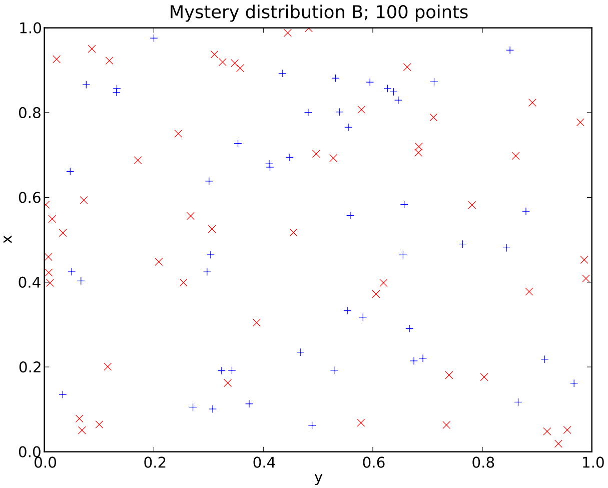


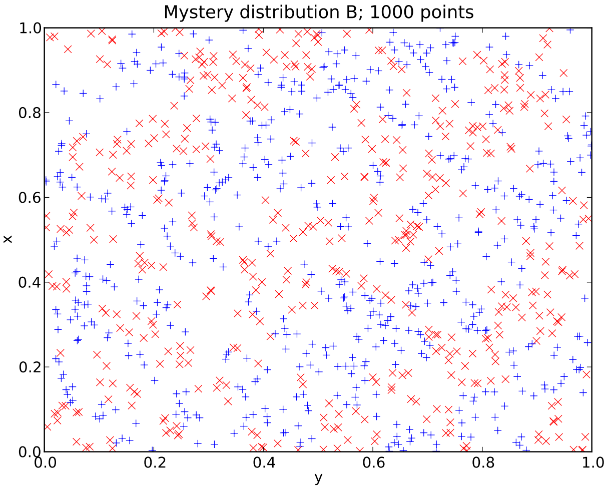
Add two features:

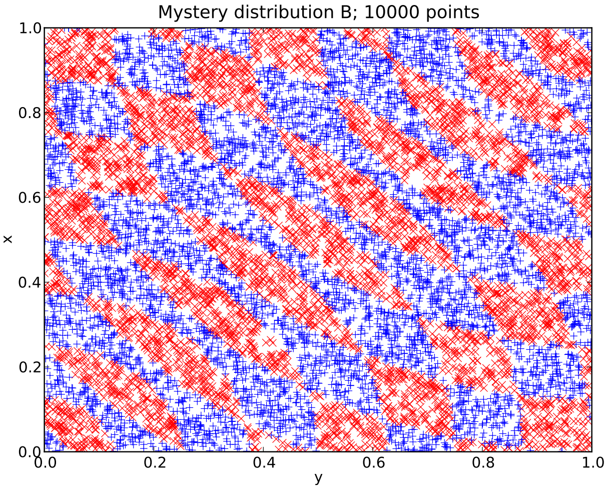
$x' = 8x \mod 2$

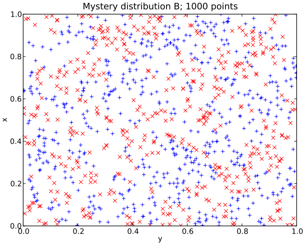
$y' = 8y \mod 2$

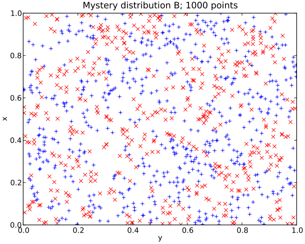

Mystery distribution B; 10 points
x
y
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0


Mystery distribution B; 100 points
x
y
0.0
0.2
0.4
0.6
0.8
1.0

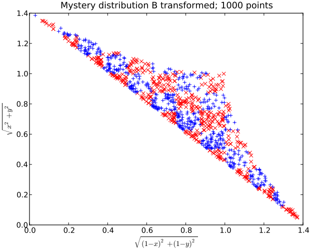

Mystery distribution B; 1000 points
1.0
0.8
0.6
0.4
0.2
0.0
x
0.0
0.2
0.4
0.6
0.8
1.0
y

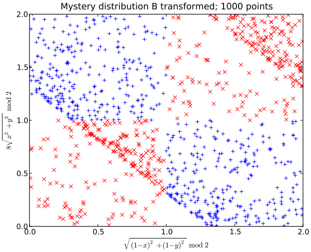

Mystery distribution B; 10000 points
1.0
0.8
0.6
0.4
0.2
0.0
x
0.0
0.2
0.4
0.6
0.8
1.0
y


Mystery distribution B; 1000 points
x
y
0.0
0.2
0.4
0.6
0.8
1.0



$\Rightarrow$





Added features:

- $d_1 = \sqrt{x^2 + y^2}$
- $d_2 = \sqrt{(1-x)^2 + (1-y)^2}$
- $m_1 = 8d_1 \mod 8$
- $m_2 = 8d_2 \mod 8.$

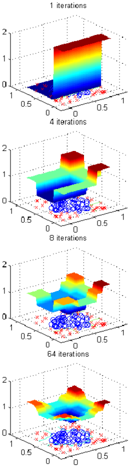
Ensemble Methods

- Decision Trees are not very useful by themselves
- Excellent basis for ensemble methods
- Use Boosting, Bagging, random sampling
- Random Forests of 10,000 decision trees
  - Better noise rejection
  - Higher capacity
  - More robust

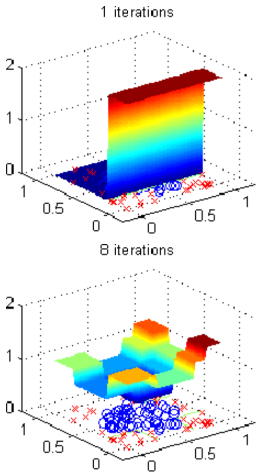
My Toolbox

- Feature Engineering
  - Mathematical Modelling
  - Smoothing
    - Principal Component Analysis
    - Deep Neural Networks
    - Auto-encoders
- Data
  - Data visualization and exploration (TSNE)
  - Data cleanup
- Machine Learning
  - Neural Nets
  - Generalized Linear Models
  - Decision trees (random forests)
  - Boosting, bagging, randomization
  - Exploration tools (explain algorithms' responses)

Boosting Algorithm



- Build on a weak learning algorithm (eg, Decision Trees)
  - it may have characteristics we want to preserve
- Use the weak learner multiple times
- Boost the weight of hard examples each time
- Robust to overfitting


1 iterations
4 iterations
8 iterations
64 iterations
2
1
0
1
0.5
0
0
0.5
1

Industrial Machine Learning

- Up to 90% of time in ETL (Extract, Transform, Load)
- Very important: data, understanding
- Important: process, features
- Least important: algorithm

When should ML be used?

- To augment modelling
- To aid in data exploration
- When you can identify what influences an outcome but not how
- When you can measure your success

When should ML not be used?

- To solve the entire problem
- To replace modelling
- To replace thought
- Without a nullable hypothesis
- When controlling something dangerous or important

Traps and Pitfalls

- Overfitting and other forms of over-optimization

Traps and Pitfalls

- Overfitting and other forms of over-optimization
- Overfitting and other forms of over-optimization

Traps and Pitfalls

- Overfitting and other forms of over-optimization
- Overfitting and other forms of over-optimization
- **Overfitting and other forms of over-optimization**

Traps and Pitfalls

- Overfitting and other forms of over-optimization
- Overfitting and other forms of over-optimization
- **Overfitting and other forms of over-optimization**
- Training on testing data
- Biasing
  - Predicting the past from the future
  - Non-representative samples
  - Multi-stage pipelines
  - Can be very subtle
- Not having a nullable hypothesis (is it really better)
- Using a complex classifier with little data
- Trying to do everything with one model
- No Free Lunch principle

Story: Optimizing the Random Numbers

Story: Optimizing the Random Numbers

- Numbers go up, but the system gets worse
- Worse than a waste of time

Undesirable Characteristics of Classifiers

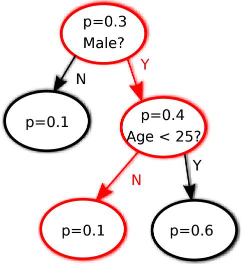
- Can't run the same experiment twice...
  - Results got worse when I retried (repeatability)
  - It took 300 CPU-years and my account was revoked (speed)
- Black box that can't tell us how to improve (explorability)
- Needs $10 million of data to learn a non-planar decision surface (non-linearity)
- Lots of knobs to play with
  - No automatic capacity control (done manually)
  - Requires $\alpha_{0,0} \ldots \zeta_{23,37}$ to be set *just right*... (automatic tuning)
  - Need to perform manual feature selection experiments
- Changing feature *z* from 0.12345 to 0.12346 lead to 13% better test performance (sensitivity)
- Output is not probabilistic

Toolbox: Cleaning Data Using Boosting

- Boosted classifiers focus almost exclusively on examples with high weights
  - High-weighted examples can have 100 times more weight than average
  - To improve the classifier, we only need to improve these examples (they effectively ignore the rest)
- If an example has a high boosting weight, it either:
  - Is a difficult (and so informative) example; or
  - Is mis-tagged
- Re-tag (carefully) those with high weights
- Can get a true positive rate $> 50\%$ with this technique

Toolbox: Explaining a Forest of Decision Trees

P(car crash)



Feature	Value	Score
Bias		0.3
Male	Y	+0.1
Age	45	-0.3

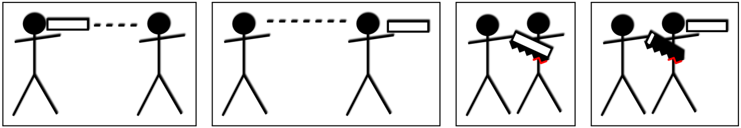
- Immensely useful in finding problems
- Given a feature vector, what features were the strongest contributors to the result?
- Record the prediction at both internal nodes and leaves
- When we follow a branch, we assign the difference between the node predictions to the splitting feature
- For multiple trees, we take the weighted sum

Application: Word Sense Disambiguation (Idilia)

- An important computational linguistics problem
  - Needed to reliably solve information retrieval, machine translation, speech recognition, ...
- Identify the meaning of words in context
  - "I enjoy a hot java in the afternoon" $\rightarrow$ coffee
  - "The economy of Java lags that of Indonesia" $\rightarrow$ island
  - "Weka is written in Java" $\rightarrow$ programming language
  - BUT: Do Java programmers in Java drink java?

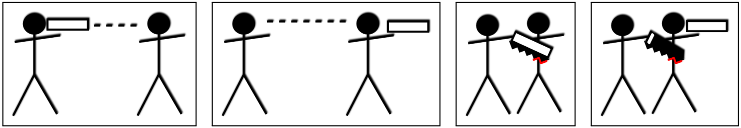
Different Types of Ambiguity

- I saw the man with the telescope.

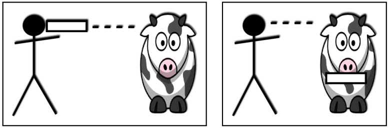


Different Types of Ambiguity

- I saw the man with the telescope.

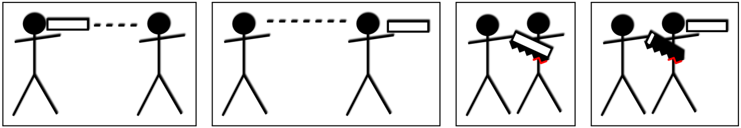


- I saw the cow with the telescope.

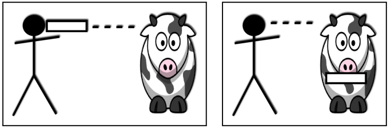


Different Types of Ambiguity

- I saw the man with the telescope.



- I saw the cow with the telescope.



- Different types of ambiguity; different ways of resolving
- Sometimes not possible to know (ill-posed)
- Some kinds of errors are very costly

Difficulties Particular to Computational Linguistics

- Language is inherently ambiguous
- Meaning is difficult to represent precisely
- Difficulties of annotation
  - Expert humans can be $< 80\%$ accurate
- Uneven Cost of errors
  - Some errors are catastrophic; others inconsequential
- Ill-posedness
- Skewed towards common meanings
  - Uncommon meanings are just as important

Using ML for Computational Linguistics

- Designed for ML from the ground up
- $1,000,000s spent tagging data
- $1,000,000s spent buying ancillary data sources
- $1,000,000s spent improving data sources

Using ML for Computational Linguistics

- Designed for ML from the ground up
- $1,000,000s spent tagging data
- $1,000,000s spent buying ancillary data sources
- $1,000,000s spent improving data sources
- 1000s of classifiers
- Pure ML algorithms never useful

Using ML for Computational Linguistics

- Designed for ML from the ground up
- $1,000,000s spent tagging data
- $1,000,000s spent buying ancillary data sources
- $1,000,000s spent improving data sources
- 1000s of classifiers
- Pure ML algorithms never useful
- ML augments language models, doesn't replace
- Each one solves a simple problem, well
- Processing pipeline of 20 stages, each unbiased wrt input
- Random forests
- Generalized Linear Models

Story: Named Entity Classification

Khalid bin Sultan, son of the Saudi Crown Prince, with US Army Commander Norman Schwartzkopf

Application: Recommendations (Recoset)

- Goal
  - Predict which product a person is most likely to buy
  - Predict which people will buy a product

Application: Recommendations (Recoset)

- Goal
  - Predict which product a person is most likely to buy
  - Predict which people will buy a product
- Data available
  - Past purchases (sometimes)
  - Clicks and ad views (sometimes)
  - Some demographic information (sometimes)
  - Product descriptions, attributes (mostly text)

Application: Recommendations (Recoset)

- Goal
  - Predict which product a person is most likely to buy
  - Predict which people will buy a product
- Data available
  - Past purchases (sometimes)
  - Clicks and ad views (sometimes)
  - Some demographic information (sometimes)
  - Product descriptions, attributes (mostly text)
- Challenges
  - Real-time
  - Very sparse data
  - Need to make predictions for unknown people
  - Must map textual descriptions to behaviour
  - Need to model attitude and preference
  - Must respect the users' privacy

Questions