New Directions in Automated Decisioning

Marc Cohen
VP Research
Fair Isaac

August 2007

Why Automate Decisioning?

- Trends/behavior changes quickly so model lifespan is short
- Refine and complement analyst-built models
- Historic data not available or reliable
- Modeling teams are small / limited throughput
- Decision alternatives change frequently
- Quickly operationalize (no separate deployment) new decision models

How We're Addressing the Need

Research automated, online decision systems that learn from past decisions and maintain decision quality (AKA **A**daptive **A**nalytics)

- Approach
  - Understand the breadth of techniques available from academia.
  - Experiment and modify to meet specific operational requirements.
- Algorithms under research:
  - Probabilistic Table Update
  - Segmentation Tree
  - Regression
  - Lazy Learning, Nearest Neighbour
- Build scale prototypes to test on simulated data and in real problem domains

Example Business Applications

- **Marketing**: Cross/Up sell offers and use accept/reject information to update the recommendation model. Over a time, offer preference is captured and maintained, even as preferences change.
- **Collections & Recovery**: Collection agencies constantly change their preference for accounts (low value, respond to letter, etc). Use AA to match account to agency.
- **Fraud**: Fraudsters try to beat the fraud system. An adaptive system will react quickly to emerging fraudulent behaviour.
- **Call centre**: Main contact point with customers – provide right customer care decisions; adjust decisions according to information.
- **Web site**: Observe browsing patterns to catch fraud, make offers, show ads, etc. Catch changes in patterns and incorporate in model.

Probabilistic Table Look-Up

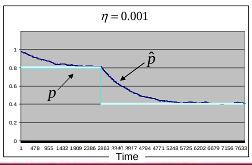
- Let i represent an offer or action choice.
- Let $\hat{p}_i(t)$ be an estimate of an event likelihood p_i at time t.

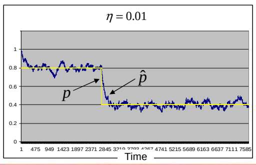
$$\hat{p}_i(t) = (1-\eta)\,\hat{p}_i(t-1) + \eta\, I_i(t)$$

η is the learning parameter

$$I_i(t) = \begin{cases} 1 & \text{with probability } p_i \\ 0 & \text{with probability } 1-p_i \end{cases}$$

- Multiple segments and multiple offers result in a table and $i^*(t) = \arg\max_i \hat{p}_i(t)$
- So the better the estimate $\hat{p}_i(t)$ the better the recommendation



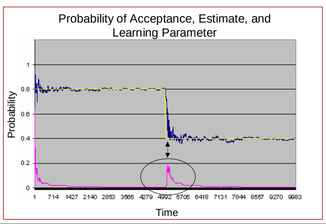


Improved Adaptive Learning

Average and sample standard deviation of 100 replications of the adaptive learning algorithm.

$$\hat{p}_i(t) = (1-\eta)\,\hat{p}_i(t-1) + \eta I_i(t)$$

$\eta \propto t$ and $|\,\hat{p}(t) - \sum_k \alpha^k \hat{p}(t-k)\,|$ learning as a function of time and estimate drift



Improved version reacts quickly to changes in the underlying model!

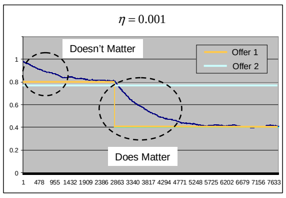
Experimentation vs. Exploitation

Slow learning means more exposure to systematic errors; fast learning means more exposure to random errors.

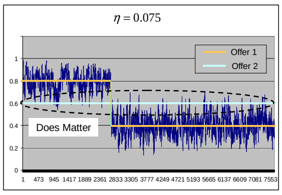
$i^*(t) = \arg\max_i \hat{p}_i(t)$ vs. improving estimate $\hat{p}_j(t)$ for $j \neq i^*(t)$

Can we characterize the precision of estimate needed for decision making?

- Multi-arm bandit problem with drift



Offer Acceptance Probability

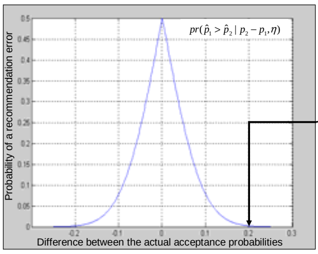


Offer Acceptance Probability

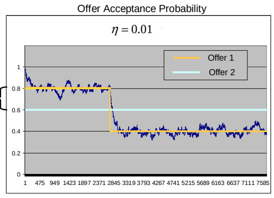
Distribution of Recommendation Errors

Estimate of a recommendation error for 2 offers as a function of the difference between the actual acceptance probabilities.

When there's no difference between the actual acceptance probabilities (at 0) then there's a 50-50 chance of making a mistake.



Offer Acceptance Probability



Assumptions: (1) normal approximation to error distribution (2) distributions are independent (3) mean and variance are not functions of probability of acceptance (4) includes initial learning and learning rate of .01.

How Does the Learning Really Work?

- How does this $i^*(t) = \arg\max_i \hat{p}_i(t)$ decision rule drive estimates of the alternative offers? --- It doesn't!
- Since we need to estimate all the alternative offer probabilities we need to make sure that alternative offers are tried. So select offer j with probability.

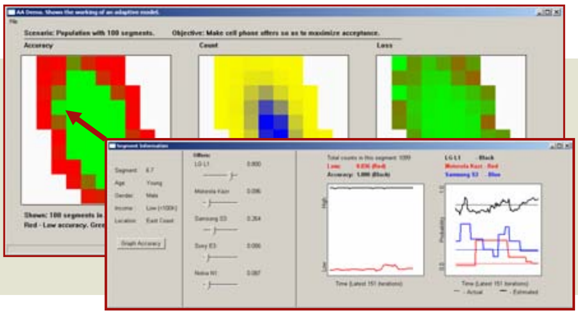
$$\frac{\hat{p}_j}{\sum_i \hat{p}_i}$$

- Or relying on the convergence of the estimate, bias the selection towards i^* by weighting the sum by the learning parameter.

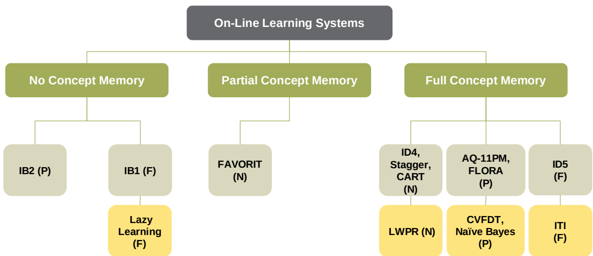
$$\frac{\hat{p}_j}{\hat{p}_{i^*} + \sum_{i \neq i^*} \hat{p}_i \eta^* \gamma}$$ ← Scaling parameter

- We could get smarter and use the likelihood of error estimates (see earlier slide) to drive the decision to try alternatives or even better use the cost of "good" and "bad" decisions.

Research/Demo Platform



Algorithm Landscape



F – Full Instance Memory
P – Partial Instance Memory
N – No Instance Memory

LWPR – Locally Weighted Project Regression
CVFDT – Concept Adapting Very Fast Decision Trees
ITI – Incremental Tree Induction

Some Classifier Algorithm Results

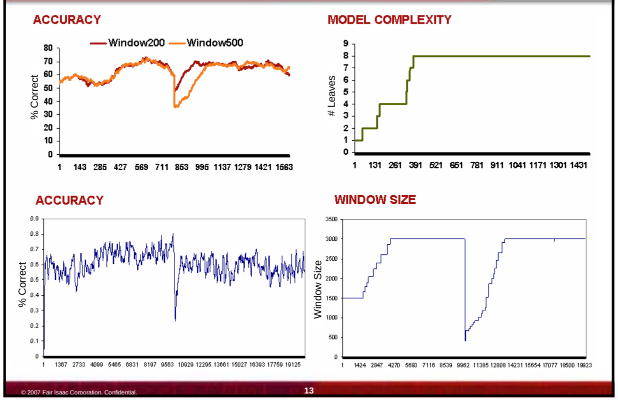
Incremental Decision Tree Induction

- Performs well with small data sets
- Scalability issues – stores all data
  - Aggravated with continuous variables
- Windowing concepts difficult to apply
- Useful where
  - Minimal/no data is available
  - No significant trend shift is expected
  - Learning can be switched off after certain period

Concept-adapting Very Fast Decision Trees

- Works in batch mode – traditional & incremental mode
- Uses sufficient statistics and starts alternate sub trees at nodes
  - Maintains window of examples
  - Checks split validity (each node) at regular intervals – is there a better splitting variable and value?
  - If any node is deemed invalid, alternate sub tree is started from that node with new split criteria
  - When alternate tree becomes better than original tree, replacement occurs – tree restructured

Concept-Adapting Very Fast DT
Results

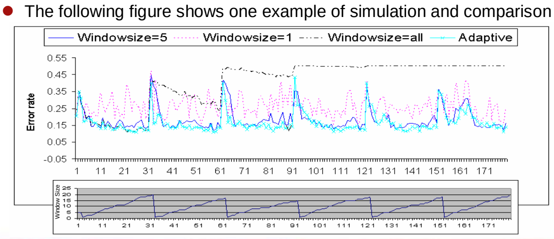


Adaptive Nearest Neighbor

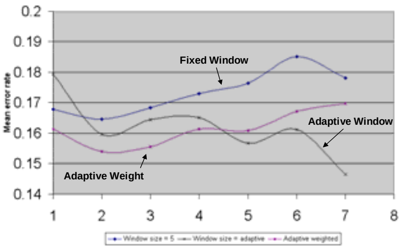
- Adaptive strategy
  - Basic idea: dynamically updating the training dataset by including new relevant data and excluding obsolete data
  - Implementation
    - Adaptively control the window size of training data
    - Weight more recent observations with adaptive exponential decay
- Nearest neighbor calculation
  - Approximate nearest neighbor (ANN) method
    - Enables approximate nearest neighbor searching in high dimensions with large amount of training data – 100-200 variables
    - Achieve significantly faster running times and large scale with relatively small errors

Adaptive Nearest Neighbor
Results

- With the adaptive strategy, the model can detect and adapt to the change of underlying structure
- The adaptive method performs better than non-adaptive (i.e., fixed window size) method in terms of both mean (i.e., average accuracy) and variance (i.e., stability) of error rate.
- The following figure shows one example of simulation and comparison



Adaptive Window vs. Adaptive Weights

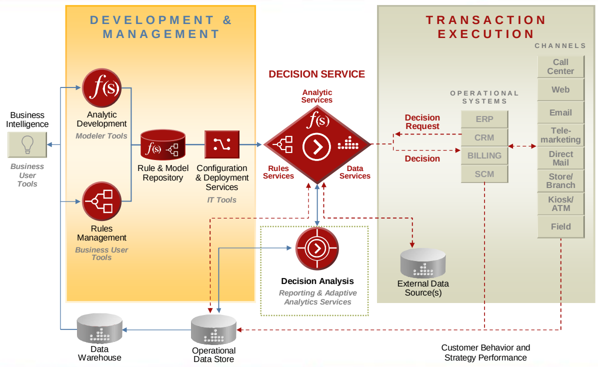


Case	1	2	3	4	5	6	7
Step Size	1	4	10	20	30	50	60
Period	.5	1	3	5	8	15	30

Continuous Drift Discrete Jumps

 16

Architecture for Enterprise Decision Management



 17

FairIsaac.

THANK YOU

Marc Cohen
415-446-6268
MarcCohen@fairisaac.com

Deena Narayanaswamy
Zhenyu Yan