Prediction Cubes

Bee-Chung Chen, Lei Chen, Yi Lin and Raghu Ramakrishnan
University of Wisconsin - Madison
Big Picture

- We are **not** trying to build a **single** accuracy “model”
- We want to find **interesting subsets** of the dataset
  - Interestingness: Defined by the “model” built on a subset
  - Cube space: A combination of dimension attribute values defines a candidate subset (just like regular OLAP)
- We are **not** using regular **aggregate functions** as the measures to summarize subsets
- We want the measures to represent **decision/prediction behavior**
  - Summarize a subset using the “model” built on it
  - Big difference from regular OLAP!!
One Sentence Summary

• Take OLAP data cubes, and keep everything the same except that we change the meaning of the cell values to represent the decision/prediction behavior
  – The idea is simple, but it leads to interesting and promising data mining tools
Example (1/5): Regular OLAP

Goal: Look for patterns of unusually high numbers of applications

Z: Dimensions
- Location
- Time
- # of App.

Y: Measure
- Cell value: Number of loan applications

<table>
<thead>
<tr>
<th>Location</th>
<th>Time</th>
<th># of App.</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL, USA</td>
<td>Dec, 04</td>
<td>2</td>
</tr>
<tr>
<td>WY, USA</td>
<td>Dec, 04</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Location</th>
<th>Time</th>
<th>2004</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA</td>
<td>Jan</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>Dec</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>50</td>
</tr>
<tr>
<td>USA</td>
<td>Jan</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td>Dec</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
</tr>
</tbody>
</table>

Roll up

Drill down

Drill down
Example (2/5): Decision Analysis

Goal: Analyze a bank’s loan decision process w.r.t. two dimensions: Location and Time

Fact table $\mathbf{D}$

<table>
<thead>
<tr>
<th>Location</th>
<th>Time</th>
<th>Race</th>
<th>Sex</th>
<th>...</th>
<th>Approval</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL, USA</td>
<td>Dec, 04</td>
<td>White</td>
<td>M</td>
<td>...</td>
<td>Yes</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>WY, USA</td>
<td>Dec, 04</td>
<td>Black</td>
<td>F</td>
<td>...</td>
<td>No</td>
</tr>
</tbody>
</table>

Model $h(X, \sigma_Z(\mathbf{D}))$

E.g., decision tree

Cube subset

Location

Time
Example (3/5): Questions of Interest

• Goal: Analyze a bank’s loan decision process with respect to two dimensions: Location and Time
• Target: Find discriminatory loan decision
• Questions:
  – Are there locations and times when the decision making was similar to a set of discriminatory decision examples (or similar to a given discriminatory decision model)?
  – Are there locations and times during which Race or Sex is an important factor of the decision process?
Example (4/5): Prediction Cube

1. Build a model using data from USA in Dec., 1985
2. Evaluate that model

Measure in a cell:
- **Accuracy** of the model
- **Predictiveness** of Race measured based on that model
- **Similarity** between that model and a given model

Model \( h(X, \sigma_{[USA, Dec 04]}(D)) \)

E.g., decision tree

Data \( \sigma_{[USA, Dec 04]}(D) \)
Example (5/5): Prediction Cube

<table>
<thead>
<tr>
<th></th>
<th>2004</th>
<th></th>
<th>2003</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan</td>
<td>Dec</td>
<td>Jan</td>
<td>Dec</td>
<td></td>
</tr>
<tr>
<td>CA</td>
<td>0.4</td>
<td>0.1</td>
<td>0.3</td>
<td>0.6</td>
</tr>
<tr>
<td>USA</td>
<td>0.7</td>
<td>0.4</td>
<td>0.3</td>
<td>0.3</td>
</tr>
</tbody>
</table>

Cell value: Predictiveness of Race

Drill down

Roll up

<table>
<thead>
<tr>
<th></th>
<th>2004</th>
<th>2003</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan</td>
<td>Dec</td>
<td>Jan</td>
<td>Dec</td>
</tr>
<tr>
<td>CA</td>
<td>0.3</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>USA</td>
<td>0.2</td>
<td>0.3</td>
<td></td>
</tr>
</tbody>
</table>

...
Outline

• Motivating example
• Definition of prediction cubes
• Efficient prediction cube materialization
• Experimental results
• Conclusion
Prediction Cubes

• User interface: OLAP data cubes
  – Dimensions, hierarchies, roll up and drill down

• Values in the cells:
  – Accuracy → Test-set accuracy cube
  – Similarity → Model-similarity cube
  – Predictiveness → Predictiveness cube
Test-Set Accuracy Cube

Given:
- Data table \(\mathbf{D}\)
- Test set \(\Delta\)

Data table \(\mathbf{D}\)

<table>
<thead>
<tr>
<th>Location</th>
<th>Time</th>
<th>Race</th>
<th>Sex</th>
<th>…</th>
<th>Approval</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL, USA</td>
<td>Dec, 04</td>
<td>White</td>
<td>M</td>
<td>…</td>
<td>Yes</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>WY, USA</td>
<td>Dec, 04</td>
<td>Black</td>
<td>F</td>
<td>…</td>
<td>No</td>
</tr>
</tbody>
</table>

The decision model of **USA during Dec 04** had high accuracy when applied to \(\Delta\).
Model-Similarity Cube

**Given:**
- Data table $D$
- Target model $h_0(X)$
- Test set $\Delta$ w/o labels

**Data table $D$**

<table>
<thead>
<tr>
<th>Location</th>
<th>Time</th>
<th>Race</th>
<th>Sex</th>
<th>...</th>
<th>Approval</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL, USA</td>
<td>Dec, 04</td>
<td>White</td>
<td>M</td>
<td>...</td>
<td>Yes</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>WY, USA</td>
<td>Dec, 04</td>
<td>Black</td>
<td>F</td>
<td>...</td>
<td>No</td>
</tr>
</tbody>
</table>

**Level:** $[\text{Country}, \text{Month}]$

The loan decision process in **USA during Dec 04** was similar to a discriminatory decision model $h_0(X)$.
Predictiveness Cube

Given:
- Data table $D$
- Attributes $V$
- Test set $\Delta$ w/o labels

Data table $D$

<table>
<thead>
<tr>
<th>Location</th>
<th>Time</th>
<th>Race</th>
<th>Sex</th>
<th>…</th>
<th>Approval</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL, USA</td>
<td>Dec, 04</td>
<td>White</td>
<td>M</td>
<td>…</td>
<td>Yes</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>WY, USA</td>
<td>Dec, 04</td>
<td>Black</td>
<td>F</td>
<td>…</td>
<td>No</td>
</tr>
</tbody>
</table>

Level: [Country, Month]

Predictiveness of $V$

Race was an important factor of loan approval decision in USA during Dec 04
Outline

- Motivating example
- Definition of prediction cubes
  - Efficient prediction cube materialization
- Experimental results
- Conclusion
One Sentence Summary

• Reduce prediction cube computation to data cube computation
  – Somehow represent a data-mining model as a distributive or algebraic (bottom-up computable) aggregate function, so that data-cube techniques can be directly applied
Full Materialization

Full Materialization Table

<table>
<thead>
<tr>
<th>Level</th>
<th>Location</th>
<th>Time</th>
<th>Cell Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>[All, All]</td>
<td>ALL</td>
<td>ALL</td>
<td>0.7</td>
</tr>
<tr>
<td>[Country, All]</td>
<td>CA</td>
<td>ALL</td>
<td>0.4</td>
</tr>
<tr>
<td>[Country, All]</td>
<td>USA</td>
<td>ALL</td>
<td>0.9</td>
</tr>
<tr>
<td>[All, Year]</td>
<td>ALL</td>
<td>1985</td>
<td>0.8</td>
</tr>
<tr>
<td>[All, Year]</td>
<td>ALL</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>[All, Year]</td>
<td>ALL</td>
<td>2004</td>
<td>0.3</td>
</tr>
<tr>
<td>[Country, Year]</td>
<td>CA</td>
<td>1985</td>
<td>0.9</td>
</tr>
<tr>
<td>[Country, Year]</td>
<td>CA</td>
<td>1986</td>
<td>0.2</td>
</tr>
<tr>
<td>[Country, Year]</td>
<td>USA</td>
<td>2004</td>
<td>0.8</td>
</tr>
</tbody>
</table>
Bottom-Up Data Cube Computation

Cell Values: Numbers of loan applications
Functions on Sets

- Bottom-up computable functions: Functions that can be computed using only summary information
  - **Distributive** function: $\alpha(X) = F(\{\alpha(X_1), \ldots, \alpha(X_n)\})$
    - $X = X_1 \cup \ldots \cup X_n$ and $X_i \cap X_j = \emptyset$
    - E.g., $\text{Count}(X) = \text{Sum}(\{\text{Count}(X_1), \ldots, \text{Count}(X_n)\})$
  - **Algebraic** function: $\alpha(X) = F(\{G(X_1), \ldots, G(X_n)\})$
    - $G(X_i)$ returns a length-fixed vector of values
    - E.g., $\text{Avg}(X) = F(\{G(X_1), \ldots, G(X_n)\})$
      - $G(X_i) = [\text{Sum}(X_i), \text{Count}(X_i)]$
      - $F(\{[s_1, c_1], \ldots, [s_n, c_n]\}) = \text{Sum}(\{s_i\}) / \text{Sum}(\{c_i\})$
Scoring Function

- Represent a model as a function of sets.
- Conceptually, a machine-learning model \( h(X; \sigma_Z(D)) \) is a scoring function \( \text{Score}(y, x; \sigma_Z(D)) \) that gives each class \( y \) a score on test example \( x \)
  - \( h(x; \sigma_Z(D)) = \arg\max_y \text{Score}(y, x; \sigma_Z(D)) \)
  - \( \text{Score}(y, x; \sigma_Z(D)) \approx p(y \mid x, \sigma_Z(D)) \)
  - \( \sigma_Z(D) \): The set of training examples (a cube subset of \( D \))
Bottom-up Score Computation

• Key observations:
  – **Observation 1:** \( \text{Score}(y, x; \sigma_Z(D)) \) is a function of cube subset \( \sigma_Z(D) \); if it is **distributive** or **algebraic**, the data cube bottom-up technique can be directly applied.
  – **Observation 2:** Having the scores for all the test examples and all the cells is **sufficient** to compute a prediction cube.
    • Scores \( \Rightarrow \) predictions \( \Rightarrow \) cell values
    • Details depend on what each cell means (i.e., type of prediction cubes); but straightforward.
1. Build a model for each lowest-level cell
2. Compute the scores using data cube bottom-up technique
   - Ob. 1: Distributive scoring function ⇒ bottom up
3. Use the scores to compute the cell values
   - Ob. 2: Having scores ⇒ having cell values
Machine-Learning Models

• Naïve Bayes:
  – Scoring function: algebraic

• Kernel-density-based classifier:
  – Scoring function: distributive

• Decision tree, random forest:
  – Neither distributive, nor algebraic

• PBE: Probability-based ensemble (new)
  – To make any machine-learning model distributive
  – Approximation
Probability-Based Ensemble

Decision tree on [WA, 85]

PBE version of decision tree on [WA, 85]

Decision trees built on the lowest-level cells
Probability-Based Ensemble

- Scoring function:

\[ h_{PBE}(x; \sigma_s(D)) = \arg \max_y Score_{PBE}(y, x; \sigma_s(D)) \]

\[ Score_{PBE}(y, x; b_i(D)) = h(y | x; b_i(D)) \cdot g(b_i | x) \]

\[ Score_{PBE}(y, x; \sigma_s(D)) = \sum_{i \in S} \left( Score_{PBE}(y, x; b_i(D)) \right) \]

- \( h(y | x; b_i(D)) \): Model h’s estimation of \( p(y | x, b_i(D)) \)
- \( g(b_i | x) \): A model that predicts the probability that \( x \) belongs to base subset \( b_i(D) \)
Outline

• Motivating example
• Definition of prediction cubes
• Efficient prediction cube materialization
• Experimental results
• Conclusion
Experiments

• Quality of PBE on 8 UCI datasets
  – The quality of the PBE version of a model is slightly worse (0 ~ 6%) than the quality of the model trained directly on the whole training data.

• Efficiency of the bottom-up score computation technique

• Case study on demographic data
Efficiency of the Bottom-up Score Computation

- Machine-learning models:
  - **J48**: J48 decision tree
  - **RF**: Random forest
  - **NB**: Naïve Bayes
  - **KDC**: Kernel-density-based classifier

- **Bottom-up method vs. Exhaustive method**
  - PBE-J48
  - PBE-RF
  - NB
  - KDC
  - J48ex
  - RFex
  - NBex
  - KDCex
Synthetic Dataset

- Dimensions: $Z_1$, $Z_2$ and $Z_3$.

- Decision rule:

<table>
<thead>
<tr>
<th>Condition</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>When $Z_1 &gt; 1$</td>
<td>$Y = I(4X_1 + 3X_2 + 2X_3 + X_4 + 0.4X_6 &gt; 7)$</td>
</tr>
<tr>
<td>else when $Z_3 \mod 2 = 0$</td>
<td>$Y = I(2X_1 + 2X_2 + 3X_3 + 3X_4 + 0.4X_6 &gt; 7)$</td>
</tr>
<tr>
<td>else</td>
<td>$Y = I(0.1X_5 + X_1 &gt; 1)$</td>
</tr>
</tbody>
</table>
Efficiency Comparison

Using exhaustive method

Using bottom-up score computation
Take-Home Messages

• Promising exploratory data analysis paradigm:
  – Use models to identify interesting subsets
  – Concentrate only on subsets in the cube space
    • Those are meaningful subsets
  – Precompute the results
  – Provide the users with an interactive tool

• A simple way to plug “something” into cube-style analysis:
  – Try to describe/approximate “something” by a distributive or algebraic function
Related Work: Building models in OLAP

- Multi-dimensional regression [Chen, VLDB 02]
  - Goal: Detect changes of trends
  - Build linear regression models for cube cells
- Step-by-step regression in stream cube [Liu, PAKDD 03]
- Loglinear-based quasi cubes [Barbara, J. IIS 01]
  - Use loglinear model to approximately compress dense regions of a data cube
- NetCube [Margaritis, VLDB 01]
  - Build Bayes Net on the entire dataset of approximately answer count queries
Related Work: Advanced Cube-Style Analysis

• Cubegrades [Imielinski, J. DMKD 02]
  – Extend data cubes using ideas from association rules
  – How the measure changes when we rollup or drill down

• Constrained gradients in data cube [Dong, VLDB 01]
  – Find pairs of similar cell characteristics associated with big changes in measure

• User-cognizant multidimensional analysis [Sarawagi, VLDBJ 01]
  – Help users to explore the most informative unvisited regions in a data cube using max entropy principle
Questions
What are Our Assumptions?

• Machine-learning models are good approximation of the true decision/prediction model
  – Evaluate accuracy

• The size of each base subset is large enough to build a good model
  – Future work: Find the proper levels of subsets to start from

• Model properties are evaluated by test sets
  – We did not consider looking at the models themselves
Why Test Set?

- To obtain quantitative model properties, we need test set
- Questions: Why to let users to provide test sets?
- Flexibility vs. ease of use
  - Flexibility: The user can specify $p(X)$ that he/she is interested in (e.g., focus on rich people)
    - E.g., compare $p_1(Y \mid X, \sigma(D))$ with $p_2(Y \mid X, \sigma(D))$
  - Simple fix:
    - Sample test set from the dataset.
    - Cross-validation cube
Why PBE is not that good?

• If the probability estimation of the base models is correct, then PBE is optimal
• Why it is not optimal in reality?
  – The probability estimation method is not good
  – The training datasets for base models are too small
• Fix:
  – Work on the probability estimation method
  – Build models for some non-base-level cells
Feature Selection vs. Prediction Cubes

- **Feature selection:**
  - Goal: Find the best $k$ predictive attributes
  - Search space: $2^n$ ($n$: number of attributes)

- **Prediction cubes:**
  - Goal: Find interesting cube cells
  - Search space: $2^d$ ($d$: number of dimension attributes)
  - You may use accuracy cube to find predictive dimension attributes, but not is not our goal
  - For the predictiveness cube, the attributes whose predictiveness is of interest is given
Why We Need Efficient Precomputation?

• Several hours vs. several days vs. several months
• For upper level cells, if the machine learning algorithm is not scalable and we do not have a bottom-up method, we may never get the result
Backup Slides
Theoretical Comparison

- Training complexity:
  - Exhaustive: \( \sum_{[l_1,\ldots,l_d] \in \text{Levels}} (|Z_1^{l_1}| \times \ldots \times |Z_d^{l_d}| \times f_{\text{train}}(n_{[l_1,\ldots,l_d]}) ) \)
  - Bottom-up: \( |Z_1^{(1)}| \times \ldots \times |Z_d^{(1)}| \times f_{\text{train}}(n_{[1,\ldots,1]}) \)

- \( Z_1 = \text{Location} \)
  - \( Z_1^{(3)} = \text{All} \)
  - \( Z_1^{(2)} = \text{State} \)
  - \( Z_1^{(1)} = \text{City} \)
    - MA
    - WI
    - MN
    - Madison, WI
    - Green Bay, WI

- \( Z_2 = \text{Time} \)
  - \( Z_2^{(3)} = \text{All} \)
  - \( Z_2^{(2)} = \text{Year} \)
  - \( Z_2^{(1)} = \text{Month} \)
    - 85
    - 86
    - 04
    - Jan., 86
    - Dec., 86
Theoretical Comparison

- Testing complexity:
  - Exhaustive: \( \sum_{[l_1,\ldots,l_d] \in \text{Levels}} \left( |Z_1^{(l_1)}| \times \ldots \times |Z_d^{(l_d)}| \times f_{\text{test}}(n_{[l_1,\ldots,l_d]}) \right) \)
  - Bottom-up: \(|Z_1^{(1)}| \times \ldots \times |Z_d^{(1)}| \times f_{\text{train}}(n_{[1,\ldots,1]}) + \sum_{[l_1,\ldots,l_d] \in (\text{Levels} - \{[1,\ldots,1]\})} \left( |Z_1^{(l_1)}| \times \ldots \times |Z_d^{(l_d)}| \times c \right) \)
Test-Set-Based Model Evaluation

• Given a set-aside test set $\Delta$ of schema $[X, Y]$:
  
  – **Accuracy** of $h(X)$:
    • The percentage of $\Delta$ that are correctly classified
  
  – **Similarity** between $h_1(X)$ and $h_2(X)$:
    • The percentage of $\Delta$ that are given the same class labels by $h_1(X)$ and $h_2(X)$
  
  – **Predictiveness** of $V \subseteq X$: (based on $h(X)$)
    • The difference between $h(X)$ and $h(X-V)$ measured by $\Delta$; i.e., the percentage of $\Delta$ that are predicted differently by $h(X)$ and $h(X-V)$
Model Accuracy

• Test-set accuracy (TS-accuracy):
  – Given a set-aside test set $\Delta$ with schema $[X, Y]$,

$$\text{accuracy}(h(X; D) \mid \Delta) = \frac{1}{|\Delta|} \sum_{(x, y) \in \Delta} I(h(x; D) = y)$$

• $|\Delta|$: The number of examples in $\Delta$
• $I(\Psi) = 1$ if $\Psi$ is true; otherwise, $I(\Psi) = 0$

• Alternative: Cross-validation accuracy
  – This will not be discussed further!!
Model Similarity

- Prediction similarity (or distance):
  - Given a set-aside test set $\Delta$ with schema $X$:
    \[
    \text{similarity}(h_1(X), h_2(X)) = \frac{1}{|\Delta|} \sum_{x \in \Delta} I(h_1(x) = h_2(x))
    \]
    \[
    \text{distance}(h_1(X), h_2(X)) = 1 - \text{similarity}(h_1(X), h_2(X))
    \]

- Similarity between $p_{h_1}(Y \mid X)$ and $p_{h_2}(Y \mid X)$:
  \[
  KL\text{-distance} = \frac{1}{|\Delta|} \sum_{x \in \Delta} \sum_y p_{h_1}(y \mid x) \log \frac{p_{h_1}(y \mid x)}{p_{h_2}(y \mid x)}
  \]
  - $p_{h_i}(Y \mid X)$: Class-probability estimated by $h_i(X)$
Attribute Predictiveness

• Predictiveness of $V \subseteq X$: (based on $h(X)$)
  – *PD-predictiveness:*
    
    $\text{distance}(h(X), h(X - V))$
  
  – *KL-predictiveness:*
    
    $\text{KL-distance}(h(X), h(X - V))$

• Alternative:

  $\text{accuracy}(h(X)) - \text{accuracy}(h(X - V))$

  – This will not be discussed further!!
Target Patterns

- Find subset $\sigma(D)$ such that $h(X; \sigma(D))$ has high prediction accuracy on a test set $\Delta$
  - E.g., The loan decision process in 2003’s WI is similar to a set $\Delta$ of discriminatory decision examples
- Find subset $\sigma(D)$ such that $h(X; \sigma(D))$ is similar to a given model $h_0(X)$
  - E.g., The loan decision process in 2003’s WI is similar to a discriminatory decision model $h_0(X)$
- Find subset $\sigma(D)$ such that $V$ is predictive on $\sigma(D)$
  - E.g., Race is an important factor of loan approval decision in 2003’s WI
Test-Set Accuracy

• We would like to discover:
  – The loan decision process in 2003’s WI is similar to a set of problematic decision examples

• Given:
  – Data table $D$: The loan decision dataset
  – Test set $\Delta$: The set of problematic decision examples

• Goal:
  – Find subset $\sigma_{Loc,Time}(D)$ such that $h(X; \sigma_{Loc,Time}(D))$ has high prediction accuracy on $\Delta$
Model Similarity

- We would like to discover:
  - The loan decision process in 2003’s WI is similar to a problematic decision model

- Given:
  - Data table $D$: The loan decision dataset
  - Model $h_0(X)$: The problematic decision model

- Goal:
  - Find subset $\sigma_{Loc,Time}(D)$ such that $h(X; \sigma_{Loc,Time}(D))$ is similar to $h_0(X)$
Attribute Predictiveness

- We would like to discover:
  - *Race* is an **important factor** of loan approval decision in **2003’s WI**

- Given:
  - Data table $D$: The loan decision dataset
  - Attribute $V$ of interest: *Race*

- Goal:
  - Find subset $\sigma_{\text{Loc,Time}}(D)$ such that $h(X; \sigma_{\text{Loc,Time}}(D))$ is very different to $h(X - V; \sigma_{\text{Loc,Time}}(D))$
Model-Based Subset Analysis

- Given: A data table $D$ with schema $[Z, X, Y]$
  - $Z$: Dimension attributes, e.g., $\{Location, Time\}$
  - $X$: Predictor attributes, e.g., $\{Race, Sex, \ldots\}$
  - $Y$: Class-label attribute, e.g., $Approval$

Data table $D$

<table>
<thead>
<tr>
<th>Location</th>
<th>Time</th>
<th>Race</th>
<th>Sex</th>
<th></th>
<th>Approval</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AL, USA</td>
<td>Dec, 04</td>
<td>White</td>
<td>M</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WY, USA</td>
<td>Dec, 04</td>
<td>Black</td>
<td>F</td>
<td></td>
<td>No</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Model-Based Subset Analysis

\[ \sigma_{[\text{USA, Dec 04}]}(D) \]

<table>
<thead>
<tr>
<th>Location</th>
<th>Time</th>
<th>Race</th>
<th>Sex</th>
<th>...</th>
<th>Approval</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL, USA</td>
<td>Dec, 04</td>
<td>White</td>
<td>M</td>
<td>...</td>
<td>Yes</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>WY, USA</td>
<td>Dec, 04</td>
<td>Black</td>
<td>F</td>
<td>...</td>
<td>No</td>
</tr>
</tbody>
</table>

- **Goal:** To understand the relationship between \( X \) and \( Y \) on different subsets \( \sigma_Z(D) \) of data \( D \)
  - Relationship: \( p(Y \mid X, \sigma_Z(D)) \)

- **Approach:**
  - Build model \( h(X; \sigma_Z(D)) \approx p(Y \mid X, \sigma_Z(D)) \)
  - Evaluate \( h(X; \sigma_Z(D)) \)
    - Accuracy, model similarity, predictiveness
Dimension and Level

$Z_1 = \text{Location}$

$Z_1^{(3)} = \text{All}\hspace{1em}Z_1^{(2)} = \text{State}\hspace{1em}Z_1^{(1)} = \text{City}$

$Z_2 = \text{Time}$

$Z_2^{(3)} = \text{All}\hspace{1em}Z_2^{(2)} = \text{Year}\hspace{1em}Z_2^{(1)} = \text{Month}$

$[3,3]$ $[2,3]$ $[3,2]$ $[1,3]$ $[2,2]$ $[3,1]$ $[1,2]$ $[2,1]$ $[1,1]$

$[\text{State},\text{All}]$ $[\text{All},\text{Year}]$ $[\text{State},\text{Year}]$ $[\text{All},\text{Month}]$ $[\text{City},\text{Year}]$ $[\text{State},\text{Month}]$ $[\text{City},\text{Month}]$
Example: Full Materialization
Scoring Function

- Conceptually, a machine-learning model $h(X; S)$ is a scoring function $Score(y, x; S)$ that gives each class $y$ a score on test example $x$
  - $h(x; S) = \arg\max_y Score(y, x; S)$
  - $Score(y, x; S) \approx p(y | x, S)$
  - $S$: A set of training examples

<table>
<thead>
<tr>
<th>Location</th>
<th>Time</th>
<th>Race</th>
<th>Sex</th>
<th>...</th>
<th>Approval</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL, USA</td>
<td>Dec, 85</td>
<td>White</td>
<td>M</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>WY, USA</td>
<td>Dec, 85</td>
<td>Black</td>
<td>F</td>
<td></td>
<td>No</td>
</tr>
</tbody>
</table>
Bottom-Up Score Computation

• Base cells: The finest-grained (lowest-level) cells in a cube
• Base subsets $b_i(D)$: The lowest-level data subsets
  – The subset of data records in a base cell is a base subset
• Properties:
  – $D = \bigcup_i b_i(D)$ and $b_i(D) \cap b_j(D) = \emptyset$
  – Any subset $\sigma_S(D)$ of $D$ that corresponds to a cube cell is the union of some base subsets
  – Notation:
    • $\sigma_S(D) = b_i(D) \cup b_j(D) \cup b_k(D)$, where $S = \{i, j, k\}$
Bottom-Up Score Computation

**Domain Lattice**

Data subset:

\[ \sigma_s(D) = \bigcup_{i \in S} b_i(D) \]

Scores:

\[ \text{Score}(y, x; \sigma_s(D)) = F(\{\text{Score}(y, x; b_i(D)) : i \in S\}) \]

<table>
<thead>
<tr>
<th>State, All</th>
<th>All, Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>[All, All]</td>
<td></td>
</tr>
<tr>
<td>[State, All]</td>
<td>[All, Year]</td>
</tr>
<tr>
<td>[State, Year]</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data subset:</th>
<th>Scores:</th>
</tr>
</thead>
<tbody>
<tr>
<td>\begin{tabular}{</td>
<td>l</td>
</tr>
</tbody>
</table>

56
Decomposable Scoring Function

- Let $\sigma_S(D) = \bigcup_{i \in S} b_i(D)$.
  - $b_i(D)$ is a base (lowest-level) subset
- Distributively decomposable scoring function:
  - $Score(y, x; \sigma_S(D)) = F(\{Score(y, x; b_i(D)) : i \in S\})$
  - $F$ is an distributive aggregate function
- Algebraically decomposable scoring function:
  - $Score(y, x; \sigma_S(D)) = F(\{G(y, x; b_i(D)) : i \in S\})$
  - $F$ is an algebraic aggregate function
  - $G(y, x; b_i(D))$ returns a length-fixed vector of values
Algorithm

• Input: The dataset \( D \) and test set \( \Delta \)
• For each lowest-level cell, which contains data \( b_i(D) \):
  – Build a model on \( b_i(D) \)
  – For each \( x \in \Delta \) and \( y \), compute:
    • \( \text{Score}(y, x; b_i(D)) \), if distributive
    • \( G(y, x; b_i(D)) \), if algebraic
• Use standard data cube computation technique to compute the scores in a bottom-up manner (by Observation 2)
• Compute the cell values using the scores (by Observation 1)
Probability-Based Ensemble

- Scoring function:

\[ h_{PBE}(x; \sigma_s(D)) = \arg \max_y \text{Score}_{PBE}(y, x; \sigma_s(D)) \]

\[ \text{Score}_{PBE}(y, x; b_i(D)) = h(y \mid x; b_i(D)) \cdot g(b_i \mid x) \]

\[ \text{Score}_{PBE}(y, x; \sigma_s(D)) = \sum_{i \in S} \left( \text{Score}_{PBE}(y, x; b_i(D)) \right) \]

- \( h(y \mid x; b_i(D)) \): Model \( h \)'s estimation of \( p(y \mid x, b_i(D)) \)
- \( g(b_i \mid x) \): A model that predicts the probability that \( x \) belongs to base subset \( b_i(D) \)
Optimality of PBE

- \( \text{Score}_{PBE}(y, x; \sigma_S(D)) = c \cdot p(y \mid x, x \in \sigma_S(D)) \)

\[
p(y \mid x, x \in \sigma_S(D)) \\
= \frac{p(y, x \in \sigma_S(D) \mid x)}{p(x \in \sigma_S(D) \mid x)} \\
= z \cdot p(y, x \in \sigma_S(D) \mid x) \\
= z \cdot \sum_{i \in S} p(y, x \in b_i(D) \mid x) \quad [ b_i(D)’s \ partitions \ \sigma_S(D)] \\
= z \cdot \sum_{i \in S} \left( p(y \mid x \in b_i(D), x) \cdot p(x \in b_i(D) \mid x) \right) \\
= z \cdot \sum_{i \in S} \left( h(y \mid x; b_i(D)) \cdot g(b_i \mid x) \right)
\]
Efficiency Comparison
Where is the Time Spend on
Accuracy of PBE

• Goal:
  – To compare PBE with the gold standard
    • PBE: A set of $n$ J48s/RFs each of which is trained on a small partition of the whole dataset
    • Gold standard: A J48/RF trained on the whole data
  – To understand how the number of base classifiers in a PBE affects the accuracy of the PBE
• Datasets:
  – Eight UCI datasets
Accuracy of PBE

Adult Dataset

Accuracy

# of base classifiers in a PBE

Accuracy of PBE
Accuracy of PBE

Nursery Dataset

Accuracy vs. # of base classifiers in a PBE

- Triangle: RF
- Square: J48
- RF-PBE
- J48-PBE
Accuracy of PBE

Error = The average of the absolute difference between a ground-truth cell value and a cell value computed by PBE