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 *Y*: Measure

Location	Time	***# of App.***
...	...	...
AL, USA	Dec, 04	2
...	...	...
WY, USA	Dec, 04	3

Coarser regions

	04	03	...
CA	100	90	...
USA	80	90	...
...	...	...	...

Roll up

	2004			2003			...
	Jan	...	Dec	Jan	...	Dec	...
CA	30	20	50	25	30	...	...
USA	70	2	8	10	...	...	...
...	...	...	...	...	...	...	...

Drill down

		2004			...
		Jan	...	Dec	...
CA	AB	20	15	15	...
	...	5	2	20	...
	YT	5	3	15	...
USA	AL	55	...	...	...
	...	5	...	...	
	WY	10	...	...	...
...	...	...	...	...	...

Cell value: Number of loan applications

Finer regions

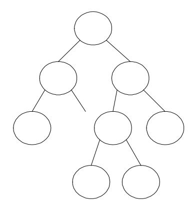
Example (2/5): Decision Analysis

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

Fact table **D**

Z: Dimensions ***X***: Predictors *Y*: Class

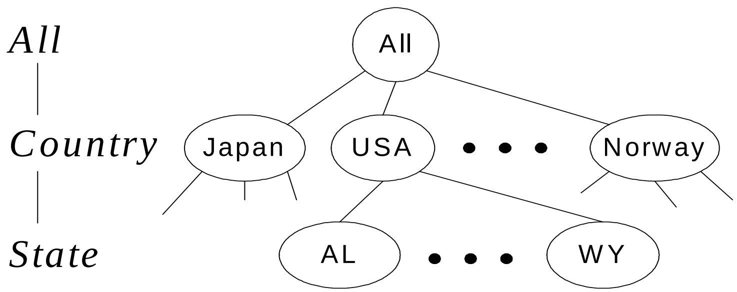
Location	Time	***Race***	***Sex***	**…**	***Approval***
AL, USA	Dec, 04	White	M	…	Yes
…	…	…	…	…	…
WY, USA	Dec, 04	Black	F	…	No



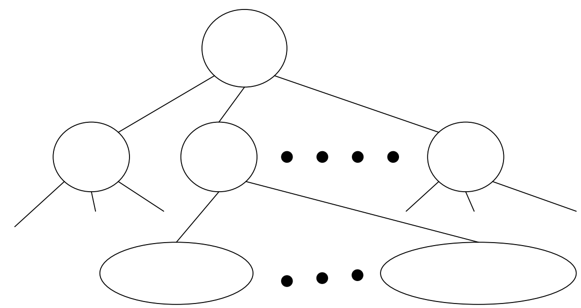
cube subset

Model $h(\boldsymbol{X}, \sigma_Z(\mathbf{D}))$
E.g., decision tree

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

	2004			2003			...
	Jan	...	Dec	Jan	...	Dec	...
CA	0.4	0.8	0.9	0.6	0.8		...
USA	0.2	0.3	0.5		...	...	...
...	...	...	...	...	...	...	...

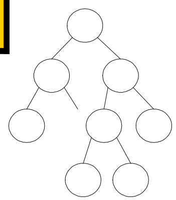
Data $\sigma_{[\mathbf{USA},\ \mathbf{Dec\ 04}]}(\mathbf{D})$

Location	Time	*Race*	*Sex*	…	*Approval*
AL ,USA	Dec, 04	White	M	…	Y
…	…	…	…	…	…
WY, USA	Dec, 04	Black	F	…	N

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 $h(\boldsymbol{X}, \sigma_{[\mathbf{USA},\ \mathbf{Dec\ 04}]}(\mathbf{D}))$
E.g., decision tree

Example (5/5): Prediction Cube

	2004			2003			...
	Jan	...	Dec	Jan	...	Dec	...
CA	0.4	0.1	0.3	0.6	0.8	...	...
USA	0.7	0.4	0.3	0.3	...	...	...
...	...	...	...	...	...	...	...

Cell value: Predictiveness of *Race*

Roll up

	04	03	...
CA	0.3	0.2	...
USA	0.2	0.3	...
...	...	...	...

Drill down

		2004			2003			...
		Jan	...	Dec	Jan	...	Dec	...
CA	AB	0.4	0.2	0.1	0.1	0.2	...	...
	...	0.1	0.1	0.3	0.3	...	...	...
	YT	0.3	0.2	0.1	0.2	...	...	...
USA	AL	0.2	0.1	0.2	...	...	...	...
	...	0.3	0.1	0.1	...	...	...	
	WY	0.9	0.7	0.8	...	...	...	...
...	...	...	...	...	...	...	...	...

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 **D**
- Test set Δ

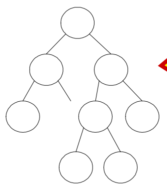
Data table **D**

Location	Time	*Race*	*Sex*	…	*Approval*
AL, USA	Dec, 04	White	M	…	Yes
…	…			…	…
WY, USA	Dec, 04	Black	F	…	No

	2004			2003			…
	Jan	…	Dec	Jan	…	Dec	…
CA	0.4	0.2	0.3	0.6	0.5	…	…
USA	0.2	0.3	0.9		…	…	…
…	…	…	…	…	…	…	…

Level: [*Country*, *Month*]

Build a model



Accuracy

Prediction

Race	*Sex*	…	*Approval*
White	F	…	Yes
…	…	…	…
Black	M	…	No

Prediction
Yes
…
Yes

Test set Δ

The decision model of **USA during Dec 04** had high accuracy when applied to Δ

Model-Similarity Cube

Given:

- Data table **D**
- Target model $h_0(\boldsymbol{X})$
- Test set Δ w/o labels

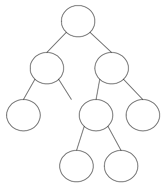
Data table **D**

Location	Time	*Race*	*Sex*	…	*Approval*
AL, USA	Dec, 04	White	M	…	Yes
…	…		…	…	…
WY, USA	Dec, 04	Black	F	…	No

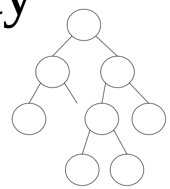
	2004			2003			…
	Jan	…	Dec	Jan	…	Dec	…
CA	0.4	0.2	0.3	0.6	0.5	…	…
USA	0.2	0.3	0.9		…	…	…
…	…	…	…	…	…	…	…

Level: [*Country*, *Month*]

Build a model



Similarity



$h_0(\boldsymbol{X})$

Race	*Sex*	…		
White	F	…	Yes	Yes
…	…	…	…	…
Black	M	…	No	Yes

Test set Δ

The loan decision process in **USA during Dec 04** was **similar to** a discriminatory decision **model**

Bee-Chung Chen, Lei Chen, Yi Lin, Raghu Ramakrishnan

Predictiveness Cube

Given:

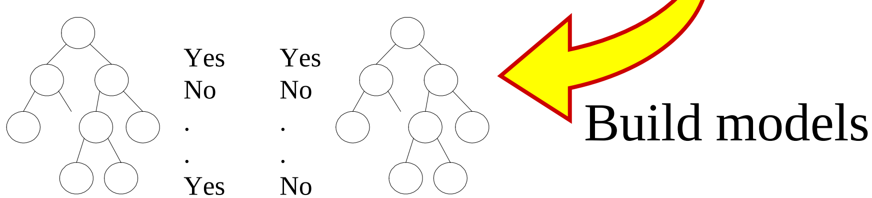
- Data table **D**
- Attributes ***V***
- Test set Δ w/o labels

Data table **D**

Location	Time	***Race***	***Sex***	**…**	***Approval***
AL, USA	Dec, 04	White	M	…	Yes
…	…	…	…	…	…
WY, USA	Dec, 04	Black	F	…	No

	2004			2003			…
	Jan	…	Dec	Jan	…	Dec	…
CA	0.4	0.2	0.3	0.6	0.5	…	…
USA	0.2	0.3	0.9		…	…	…
…	…	…	…	…	…	…	…

Level: [*Country*, *Month*]



$h(\boldsymbol{X})$ $h(\boldsymbol{X}-\boldsymbol{V})$

Predictiveness of ***V***

Race	***Sex***	**…**
White	F	…
…	…	…
Black	M	…

Test set Δ

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

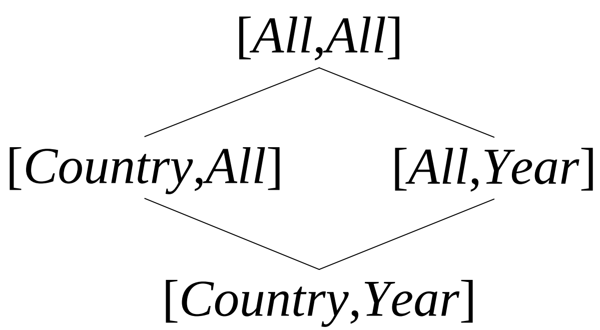
Outline

- 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



[All, Year]

	1985	1986	…	2004
All				

[All, All]

	All
All	

	1985	1986	…	2004
CA				
…				
USA				

[Country, Year]

	All
CA	
…	
USA	

[Country, All]

Full Materialization Table

Level	*Location*	*Time*	Cell Value
[All,All]	ALL	ALL	0.7
[Country,All]	CA	ALL	0.4
	…	ALL	…
	USA	ALL	0.9
[All,Year]	ALL	1985	0.8
	ALL	…	…
	ALL	2004	0.3
[Country,Year]	CA	1985	0.9
	CA	1986	0.2
	…	…	…
	USA	2004	0.8

Bottom-Up Data Cube Computation

	1985	1986	1987	1988
All	47	107	76	67

	All
All	297

	1985	1986	1987	1988
Norway	10	30	20	24
…	23	45	14	32
USA	14	32	42	11

	All
Norway	84
…	114
USA	99

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 $X_i \cap X_j = \varnothing$
  - E.g., $Count(X) = Sum(\{Count(X_1), \ldots, Count(X_n)\})$
- **Algebraic** function: $\alpha(X) = F(\{G(X_1), \ldots, G(X_n)\})$
  - $G(X_i)$ returns a length-fixed vector of values
  - E.g., $Avg(X) = F(\{G(X_1), \ldots, G(X_n)\})$
    - $G(X_i) = [Sum(X_i), Count(X_i)]$
    - $F(\{[s_1, c_1], \ldots, [s_n, c_n]\}) = Sum(\{s_i\}) / Sum(\{c_i\})$

Scoring Function

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

Bottom-up Score Computation

- Key observations:
  - **Observation 1:** $Score(y, \mathbf{x}; \sigma_Z(\mathbf{D}))$ is a function of cube subset $\sigma_Z(\mathbf{D})$; 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

	1985	1986	1987	1988
All	value	value	value	value

	All
All	value

	1985	1986	1987	1988
Norway	value	value	value	value
…	value	value	value	value
USA	value	value	value	value

	All
Norway	value
…	value
USA	value

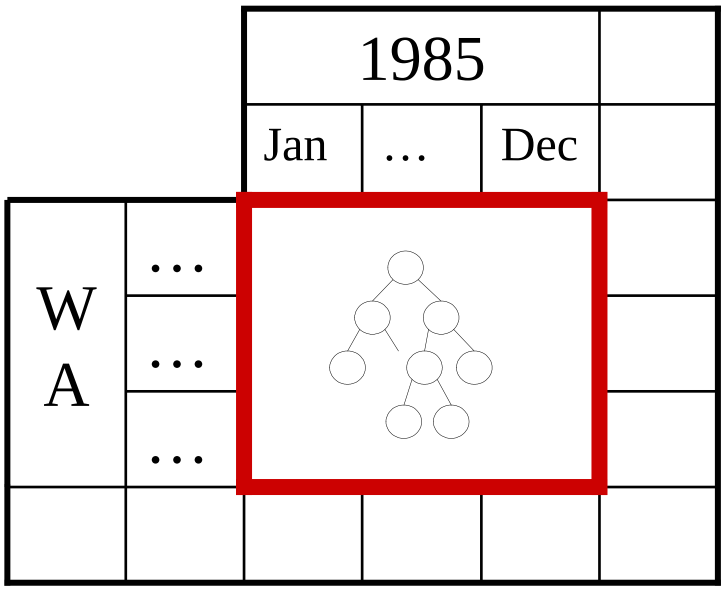
1. Build a model for each lowest-level cell
2. Compute the scores using data cube bottom-up technique
   - Ob. 1: Distributive scoring function $\Rightarrow$ bottom up
3. Use the scores to compute the cell values
   - Ob. 2: Having scores $\Rightarrow$ having cell values

Machine-Learning Models

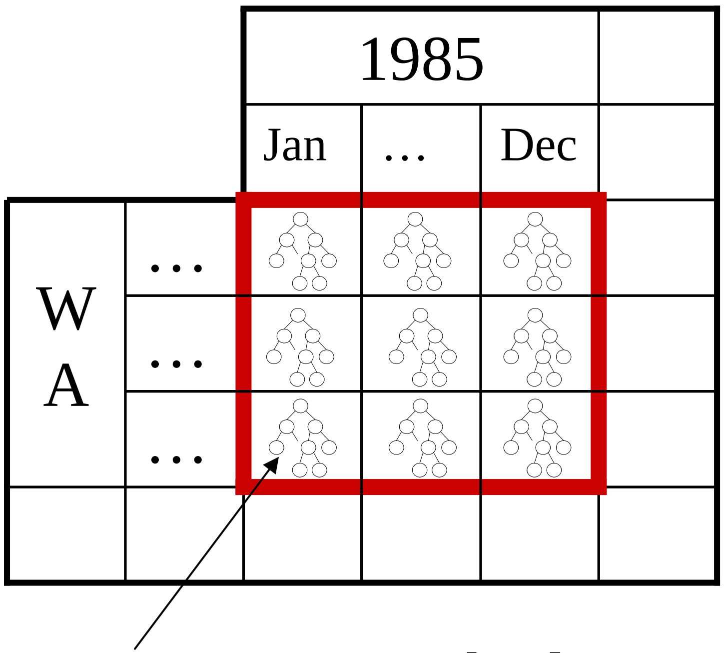
- Naïve Bayes:
  - Scoring function: algebraic
- Kernel-density-based classifier:
  - Scoring function: distributive
- Decision tree, random forest:
  - Neither distributive, nor algebraic
- PBE: Probability-based ensemble (new)
  - To make any machine-learning model distributive
  - Approximation

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:

$$h_{PBE}(\mathbf{x};\sigma_{\mathbf{S}}(\mathbf{D})) = \arg\max_y Score_{PBE}(y,\mathbf{x};\sigma_{\mathbf{S}}(\mathbf{D}))$$

$$Score_{PBE}(y,\mathbf{x};b_i(\mathbf{D})) = h(y \mid \mathbf{x};b_i(\mathbf{D})) \cdot g(b_i \mid \mathbf{x})$$

$$Score_{PBE}(y,\mathbf{x};\sigma_{\mathbf{S}}(\mathbf{D})) = \sum_{i\in\mathbf{S}} \left(Score_{PBE}(y,\mathbf{x};b_i(\mathbf{D}))\right)$$

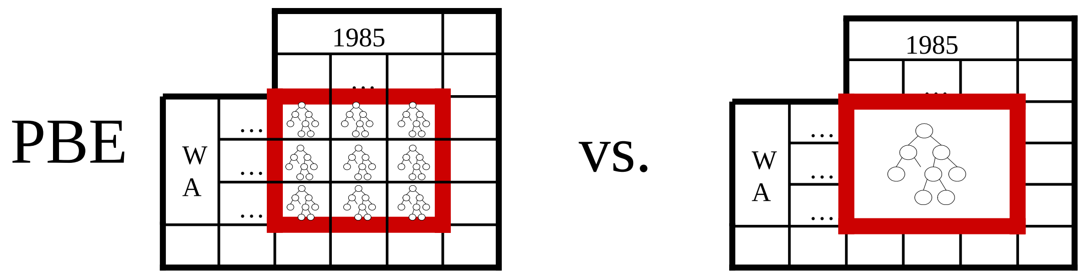
  - $h(y \mid \mathbf{x}; b_i(\mathbf{D}))$: Model h's estimation of $p(y \mid \mathbf{x}, b_i(\mathbf{D}))$
  - $g(b_i \mid \mathbf{x})$: A model that predicts the probability that $\mathbf{x}$ belongs to base subset $b_i(\mathbf{D})$

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

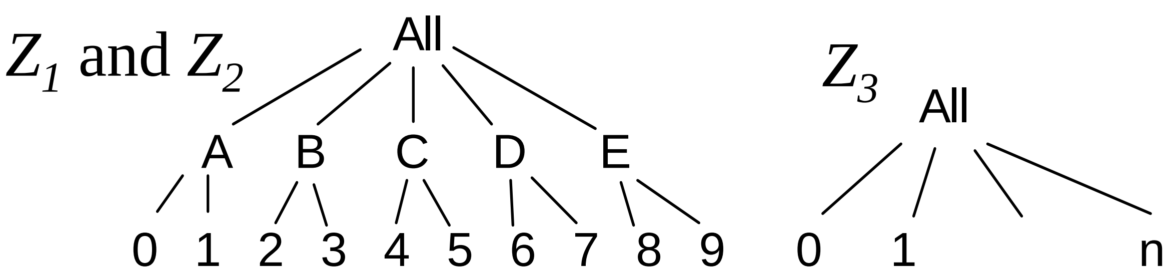
Efficiency of the Bottom-up Score Computation

- 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	– J48ex
– PBE-RF	– RFex
– NB	– NBex
– KDC	– KDCex

Synthetic Dataset

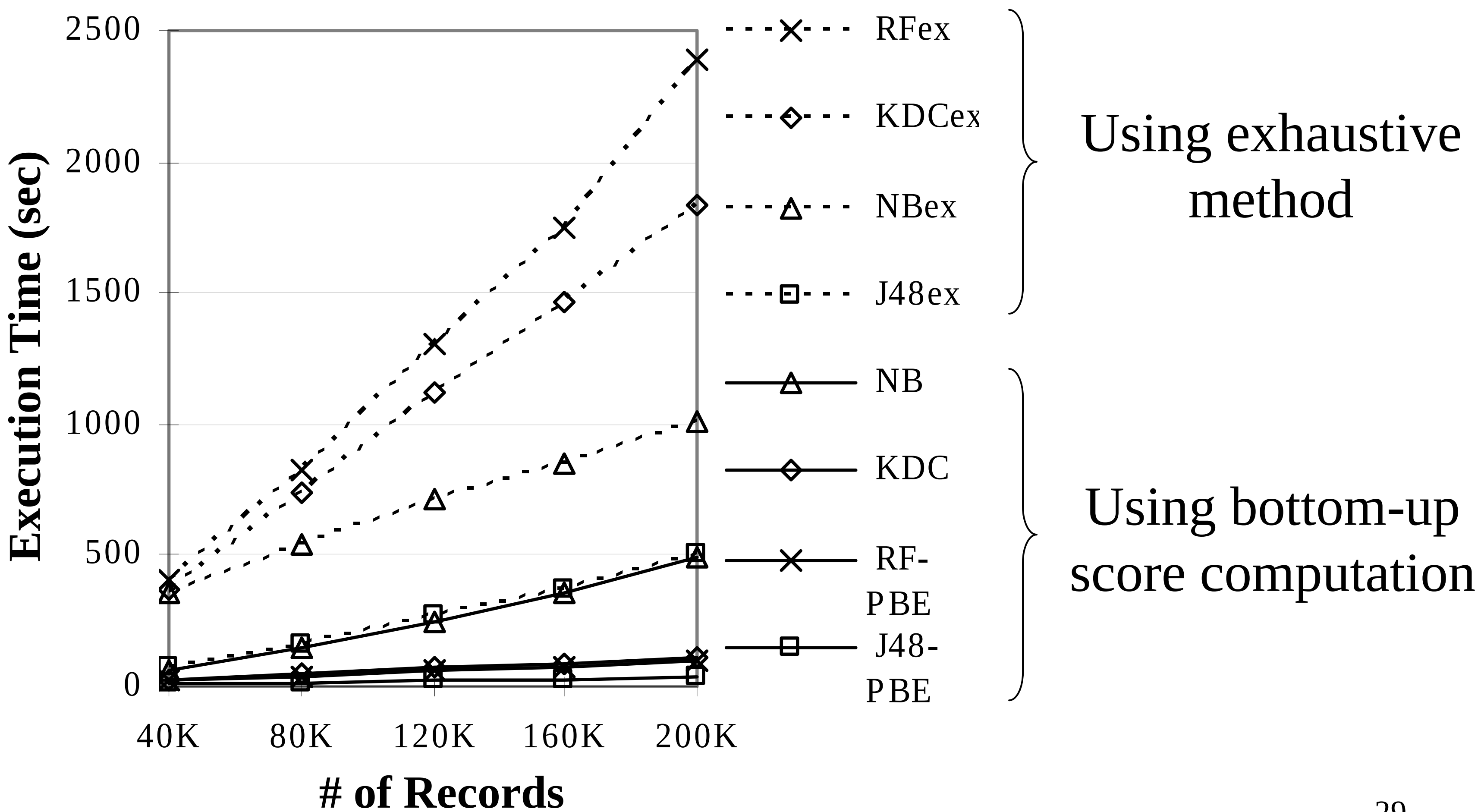
- Dimensions: Z_1, Z_2 and Z_3.



- Decision rule:

Condition	**Rule**
When Z_1>1	$Y = I(4X_1+3X_2+2X_3+X_4+0.4X_6 > 7)$
else when Z_3 mod 2 = 0	$Y = I(2X_1+2X_2+3X_3+3X_4+0.4X_6 > 7)$
else	$Y = I(0.1X_5+X_1>1)$

Efficiency Comparison



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 $p(\boldsymbol{X})$ that he/she is interested in (e.g., focus on rich people)
    - E.g., compare $p_1(Y \mid \boldsymbol{X}, \sigma(\mathbf{D}))$ with $p_2(Y \mid \boldsymbol{X}, \sigma(\mathbf{D}))$
  - 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

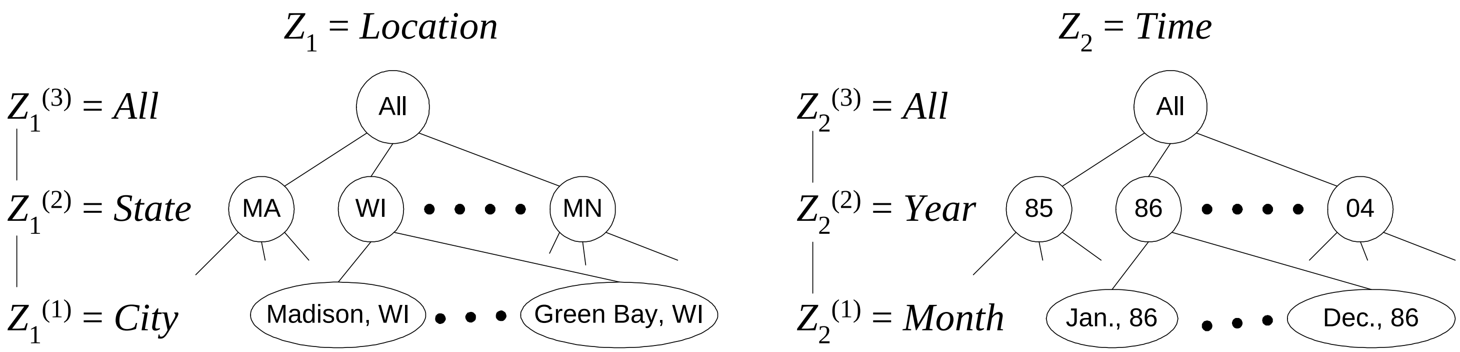
Why We Need Efficient Precomputation?

- 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

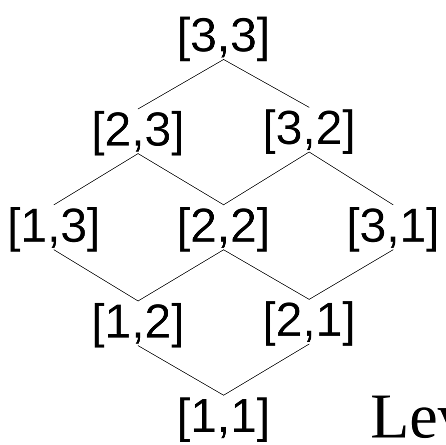
Theoretical Comparison

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



Theoretical Comparison

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



Levels

Test-Set-Based Model Evaluation

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

Model Accuracy

- Test-set accuracy (TS-accuracy):
  - Given a set-aside test set Δ with schema [***X***, *Y*],

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

    - |Δ|: The number of examples in Δ
    - $I(\Psi) = 1$ if Ψ 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 Δ with schema $\boldsymbol{X}$:

$$similarity(h_1(\boldsymbol{X}), h_2(\boldsymbol{X})) = \frac{1}{|\Delta|}\sum_{\mathbf{x}\in\Delta} I(h_1(\mathbf{x}) = h_2(\mathbf{x}))$$

$$distance(h_1(\boldsymbol{X}), h_2(\boldsymbol{X})) = 1 - \text{similarity}(h_1(\boldsymbol{X}), h_2(\boldsymbol{X}))$$

- Similarity between $p_{h_1}(Y \mid \boldsymbol{X})$ and $p_{h_2}(Y \mid \boldsymbol{X})$:

$$KL\text{-}distance = \frac{1}{|\Delta|}\sum_{\mathbf{x}\in\Delta}\sum_{y} p_{h_1}(y \mid x)\log\frac{p_{h_1}(y \mid x)}{p_{h_2}(y \mid x)}$$

  - $p_{h_i}(Y \mid \boldsymbol{X})$: Class-probability estimated by $h_i(\boldsymbol{X})$

Attribute Predictiveness

- Predictiveness of $\boldsymbol{V} \subseteq \boldsymbol{X}$: (based on $h(\boldsymbol{X})$)
  - *PD-predictiveness*:

$$\textit{distance}(h(\boldsymbol{X}), h(\boldsymbol{X} - \boldsymbol{V}))$$

  - *KL-predictiveness*:

$$\textit{KL-distance}(h(\boldsymbol{X}), h(\boldsymbol{X} - \boldsymbol{V}))$$

- Alternative:

$$\textit{accuracy}(h(\boldsymbol{X})) - \textit{accuracy}(h(\boldsymbol{X} - \boldsymbol{V}))$$

  - This will not be discussed further!!

Target Patterns

- Find subset $\sigma(\mathbf{D})$ such that $h(\boldsymbol{X}; \sigma(\mathbf{D}))$ has high prediction accuracy on a test set Δ
  - E.g., The loan decision process in 2003's WI is similar to a set Δ of discriminatory decision examples
- Find subset $\sigma(\mathbf{D})$ such that $h(\boldsymbol{X}; \sigma(\mathbf{D}))$ is similar to a given model $h_0(\boldsymbol{X})$
  - E.g., The loan decision process in 2003's WI is similar to a discriminatory decision model $h_0(\boldsymbol{X})$
- Find subset $\sigma(\mathbf{D})$ such that $\boldsymbol{V}$ is predictive on $\sigma(\mathbf{D})$
  - 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 Δ: The set of problematic decision examples
- Goal:
  - Find subset $\sigma_{Loc,Time}(\mathbf{D})$ such that $h(\boldsymbol{X}; \sigma_{Loc,Time}(\mathbf{D}))$ has high prediction accuracy on Δ

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 $h_0(\boldsymbol{X})$: The problematic decision model
- Goal:
  - Find subset $\sigma_{Loc,Time}(\mathbf{D})$ such that $h(\boldsymbol{X}; \sigma_{Loc,Time}(\mathbf{D}))$ is similar to $h_0(\boldsymbol{X})$

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 $\sigma_{Loc,Time}(\mathbf{D})$ such that $h(\boldsymbol{X}; \sigma_{Loc,Time}(\mathbf{D}))$ is very different to $h(\boldsymbol{X} - \boldsymbol{V}; \sigma_{Loc,Time}(\mathbf{D}))$

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*, …}
  - *Y*: Class-label attribute, e.g., *Approval*

Data table **D**

Location	Time	***Race***	***Sex***	**…**	***Approval***
AL, USA	Dec, 04	White	M	…	Yes
…	…	…	…	…	…
WY, USA	Dec, 04	Black	F	…	No

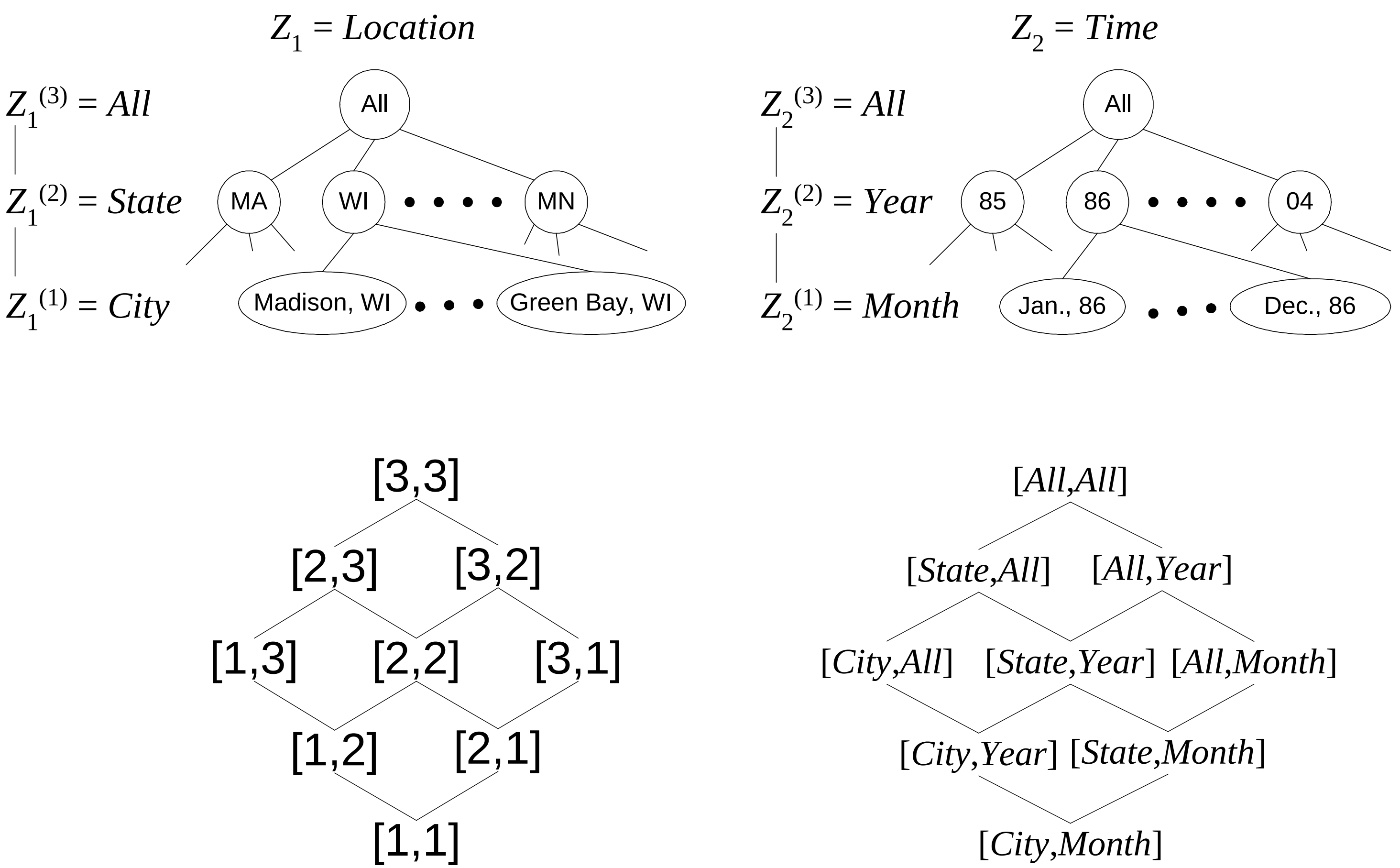
Model-Based Subset Analysis

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

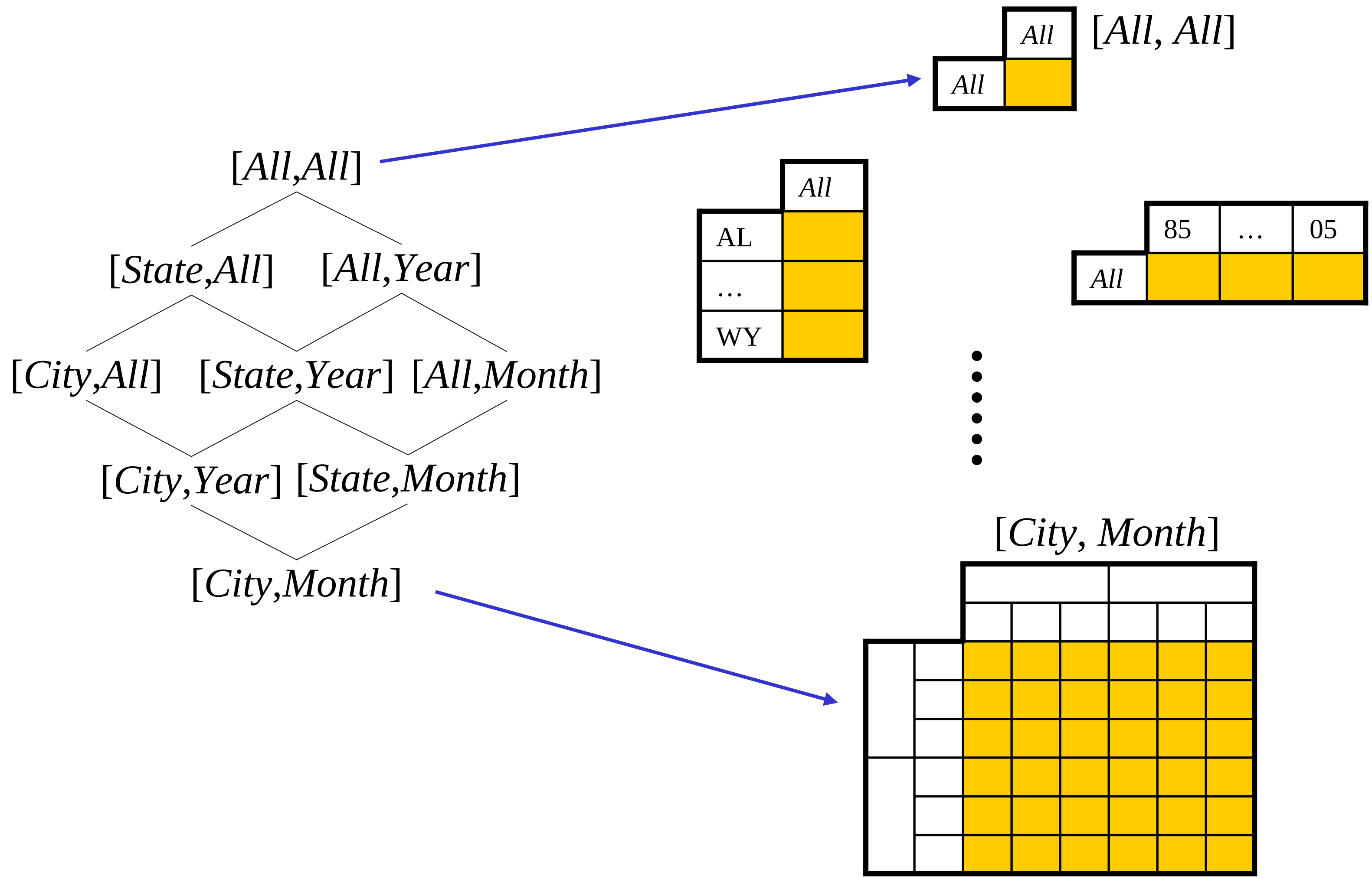
***Z*: Dimension**		***X*: Predictor**			***Y*: Class**
Location	Time	***Race***	***Sex***	**…**	***Approval***
AL, USA	Dec, 04	White	M	…	Yes
…	…	…	…	…	…
WY, USA	Dec, 04	Black	F	…	No

- Goal: To understand the relationship between $\boldsymbol{X}$ and Y on different subsets $\sigma_Z(\mathbf{D})$ of data $\mathbf{D}$
  - Relationship: $p(Y \mid \boldsymbol{X}, \sigma_Z(\mathbf{D}))$
- Approach:
  - Build model $h(\boldsymbol{X}; \sigma_Z(\mathbf{D})) \approx p(Y \mid \boldsymbol{X}, \sigma_Z(\mathbf{D}))$
  - Evaluate $h(\boldsymbol{X}; \sigma_Z(\mathbf{D}))$
    - Accuracy, model similarity, predictiveness

Dimension and Level



Example: Full Materialization

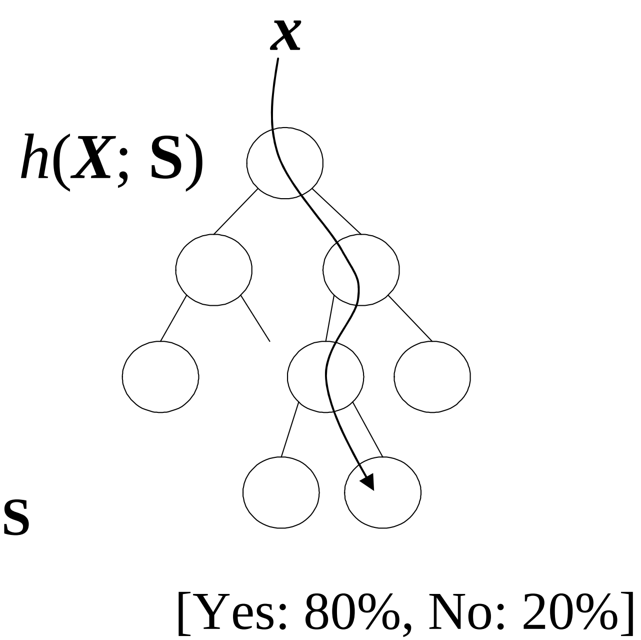


Scoring Function

- Conceptually, a machine-learning model $h(\boldsymbol{X}; \mathbf{S})$ is a scoring function $Score(y, \boldsymbol{x}; \mathbf{S})$ that gives each class y a score on test example $\boldsymbol{x}$
  - $h(\boldsymbol{x}; \mathbf{S}) = \text{argmax}_y \; Score(y, \boldsymbol{x}; \mathbf{S})$
  - $Score(y, \boldsymbol{x}; \mathbf{S}) \approx p(y \mid \boldsymbol{x}, \mathbf{S})$
  - **S**: A set of training examples

Location	Time	*Race*	*Sex*	…	*Approval*
AL, USA	Dec, 85	White	M	…	Yes
…	…	…	…	…	…
WY, USA	Dec, 85	Black	F	…	No

S

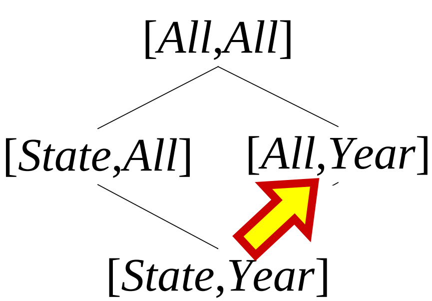


Bottom-Up Score Computation

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

Bottom-Up Score Computation

Domain Lattice



Data subset:

$\sigma_S(\mathbf{D}) = \cup_{i \in S}\, b_i(\mathbf{D})$

	1985	**…**
All	$\sigma_S(\mathbf{D})$	…

	1985	**…**
WA	$b_1(\mathbf{D})$	…
WI	$b_2(\mathbf{D})$	
WY	$b_3(\mathbf{D})$	…

Scores:

$Score(y, \mathbf{x}; \sigma_S(\mathbf{D})) =$
$F(\{Score(y, \mathbf{x}; b_i(\mathbf{D})) : i \in \boldsymbol{S}\})$

	1985	**…**
All	$Score(y, \mathbf{x}; \sigma_S(\mathbf{D}))$	…

	1985	**…**
WA	$Score(y, \mathbf{x}; b_1(\mathbf{D}))$	…
WI	$Score(y, \mathbf{x}; b_2(\mathbf{D}))$	
WY	$Score(y, \mathbf{x}; b_3(\mathbf{D}))$	…

Decomposable Scoring Function

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

Algorithm

- Input: The dataset **D** and test set Δ
- For each lowest-level cell, which contains data $b_i(\mathbf{D})$:
  - Build a model on $b_i(\mathbf{D})$
  - For each $\boldsymbol{x} \in \Delta$ and y, compute:
    - $Score(y, \boldsymbol{x}; b_i(\mathbf{D}))$, if distributive
    - $G(y, \boldsymbol{x}; b_i(\mathbf{D}))$, 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:

$$h_{PBE}(\boldsymbol{x};\sigma_{\boldsymbol{S}}(\mathbf{D})) = \arg\max_{y} Score_{PBE}(y,\boldsymbol{x};\sigma_{\boldsymbol{S}}(\mathbf{D}))$$

$$Score_{PBE}(y,\boldsymbol{x};b_i(\mathbf{D})) = h(y \mid \boldsymbol{x};b_i(\mathbf{D})) \cdot g(b_i \mid \boldsymbol{x})$$

$$Score_{PBE}(y,\boldsymbol{x};\sigma_{\boldsymbol{S}}(\mathbf{D})) = \sum_{i\in\boldsymbol{S}} \left(Score_{PBE}(y,\boldsymbol{x};b_i(\mathbf{D}))\right)$$

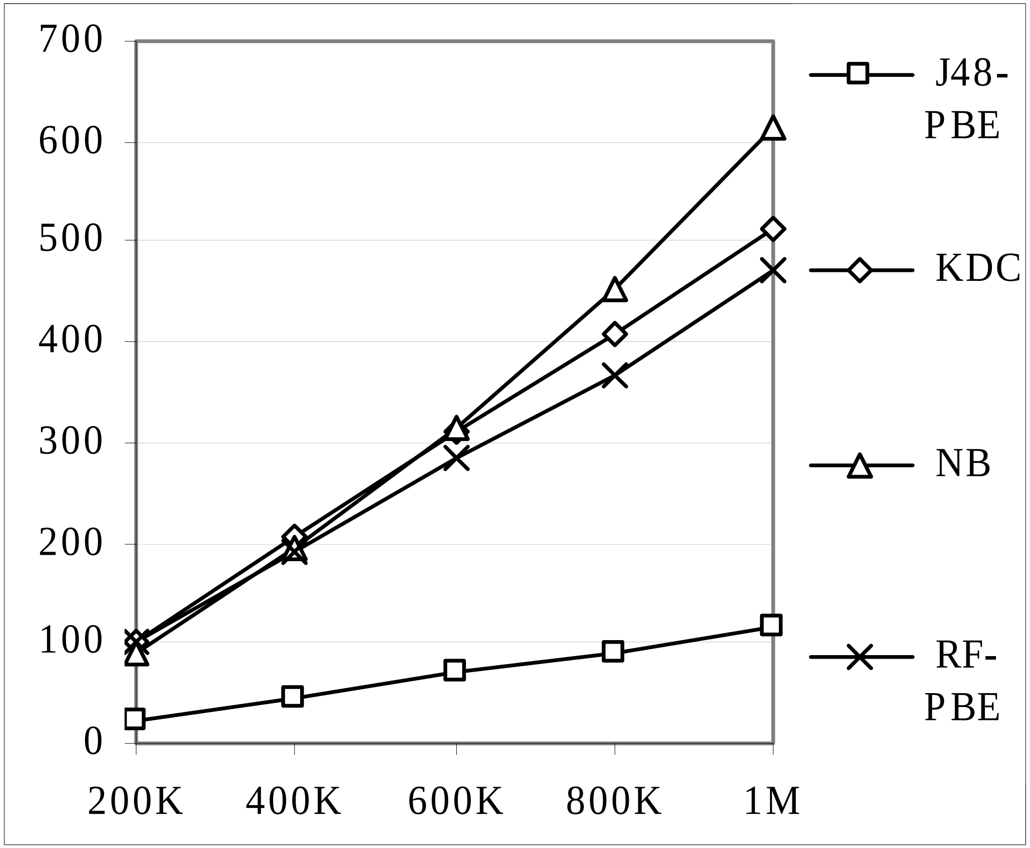
  - $h(y \mid \boldsymbol{x}; b_i(\mathbf{D}))$: Model *h*'s estimation of $p(y \mid \boldsymbol{x}, b_i(\mathbf{D}))$
  - $g(b_i \mid \boldsymbol{x})$: A model that predicts the probability that $\boldsymbol{x}$ belongs to base subset $b_i(\mathbf{D})$

Optimality of PBE

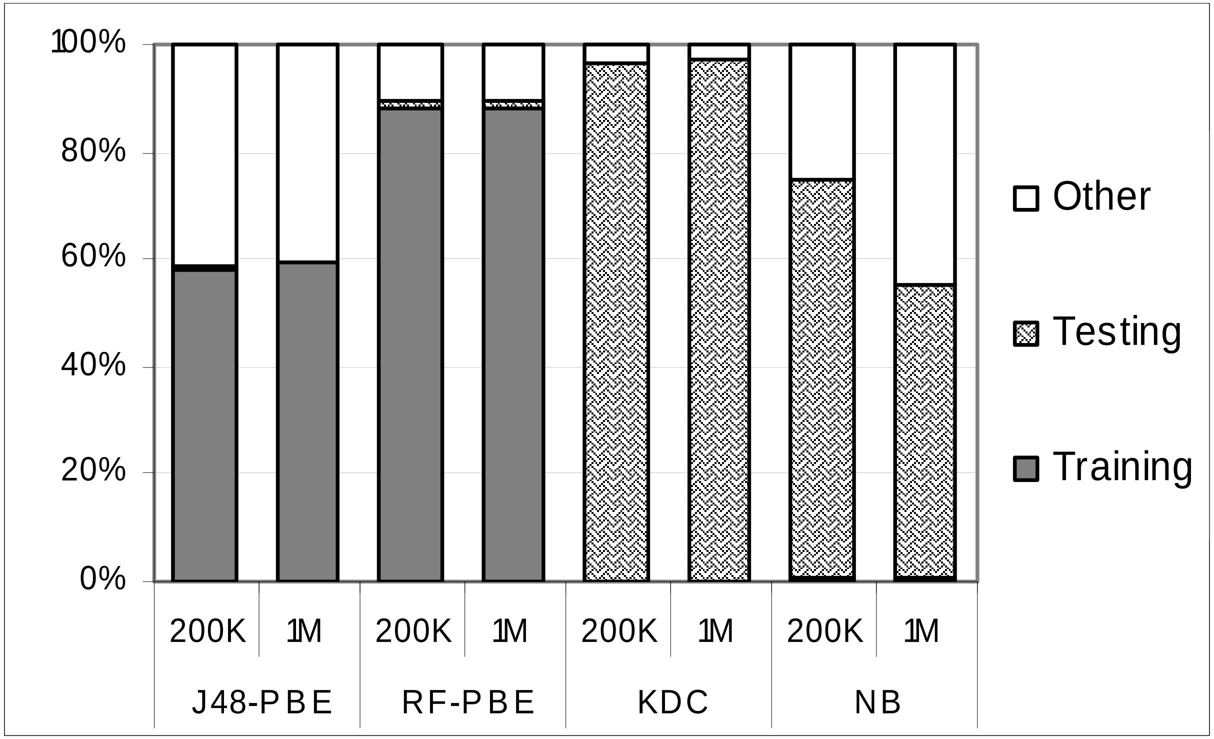
- $Score_{PBE}(y, \mathbf{x}; \sigma_{\mathbf{S}}(\mathbf{D})) = c \cdot p(y \mid \mathbf{x}, \mathbf{x} \in \sigma_{\mathbf{S}}(\mathbf{D}))$

$$
\begin{aligned}
& p(y \mid \mathbf{x}, \mathbf{x} \in \sigma_{\mathbf{S}}(\mathbf{D})) \\
& \quad = \frac{p(y, \mathbf{x} \in \sigma_{\mathbf{S}}(\mathbf{D}) \mid \mathbf{x})}{p(\mathbf{x} \in \sigma_{\mathbf{S}}(\mathbf{D}) \mid \mathbf{x})} \\
& \quad = z \cdot p(y, \mathbf{x} \in \sigma_{\mathbf{S}}(\mathbf{D}) \mid \mathbf{x}) \\
& \quad = z \cdot \sum_{i \in \mathbf{S}} p(y, \mathbf{x} \in b_i(\mathbf{D}) \mid \mathbf{x}) \qquad [\, b_i(\mathbf{D})\text{'s partitions } \sigma_{\mathbf{S}}(\mathbf{D})] \\
& \quad = z \cdot \sum_{i \in \mathbf{S}} \left(p(y \mid \mathbf{x} \in b_i(\mathbf{D}), \mathbf{x}) \cdot p(\mathbf{x} \in b_i(\mathbf{D}) \mid \mathbf{x}) \right) \\
& \quad = z \cdot \sum_{i \in \mathbf{S}} \left(h(y \mid \mathbf{x}; b_i(\mathbf{D})) \cdot g(b_i \mid \mathbf{x}) \right)
\end{aligned}
$$

Efficiency Comparison



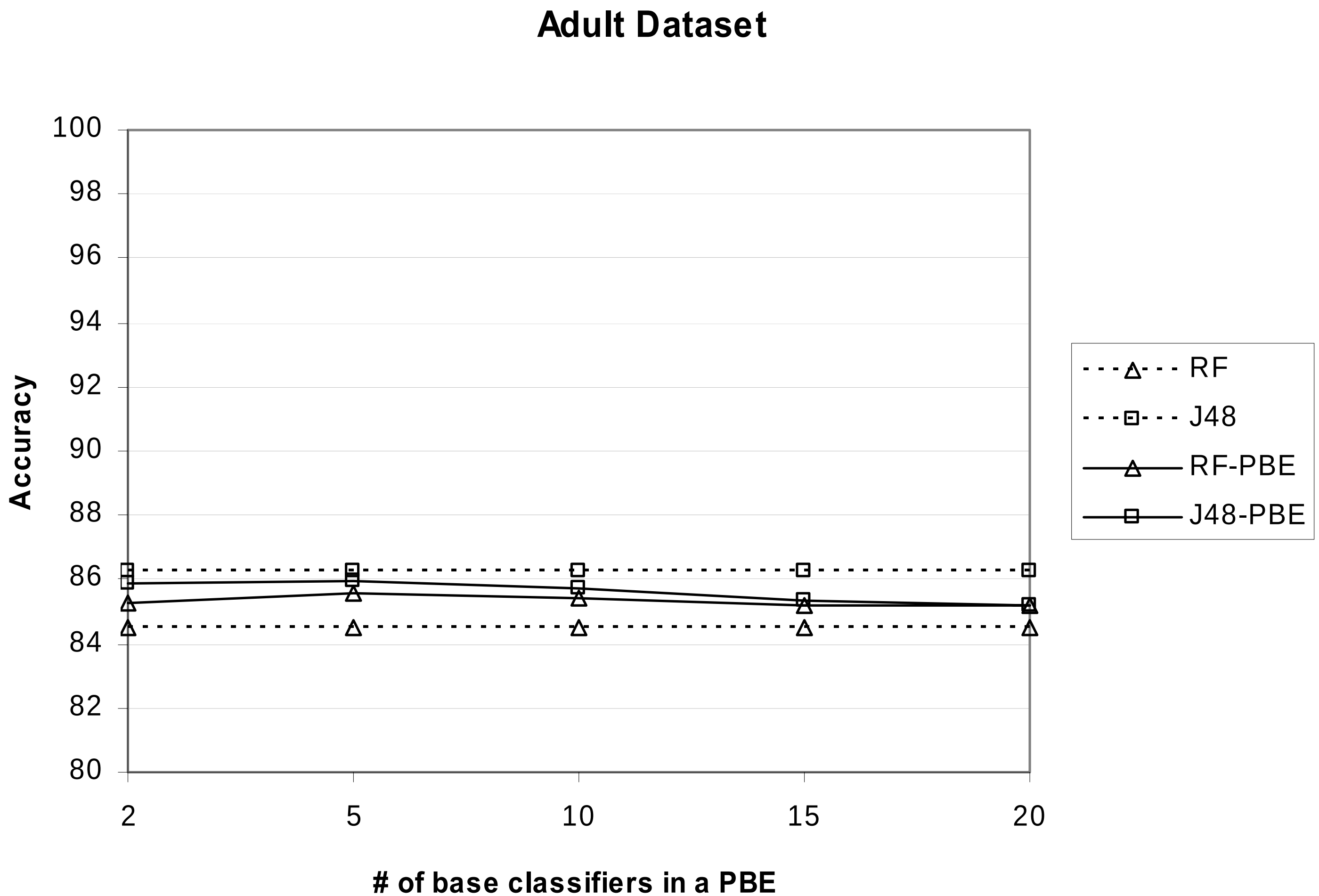
Where is the Time Spend on



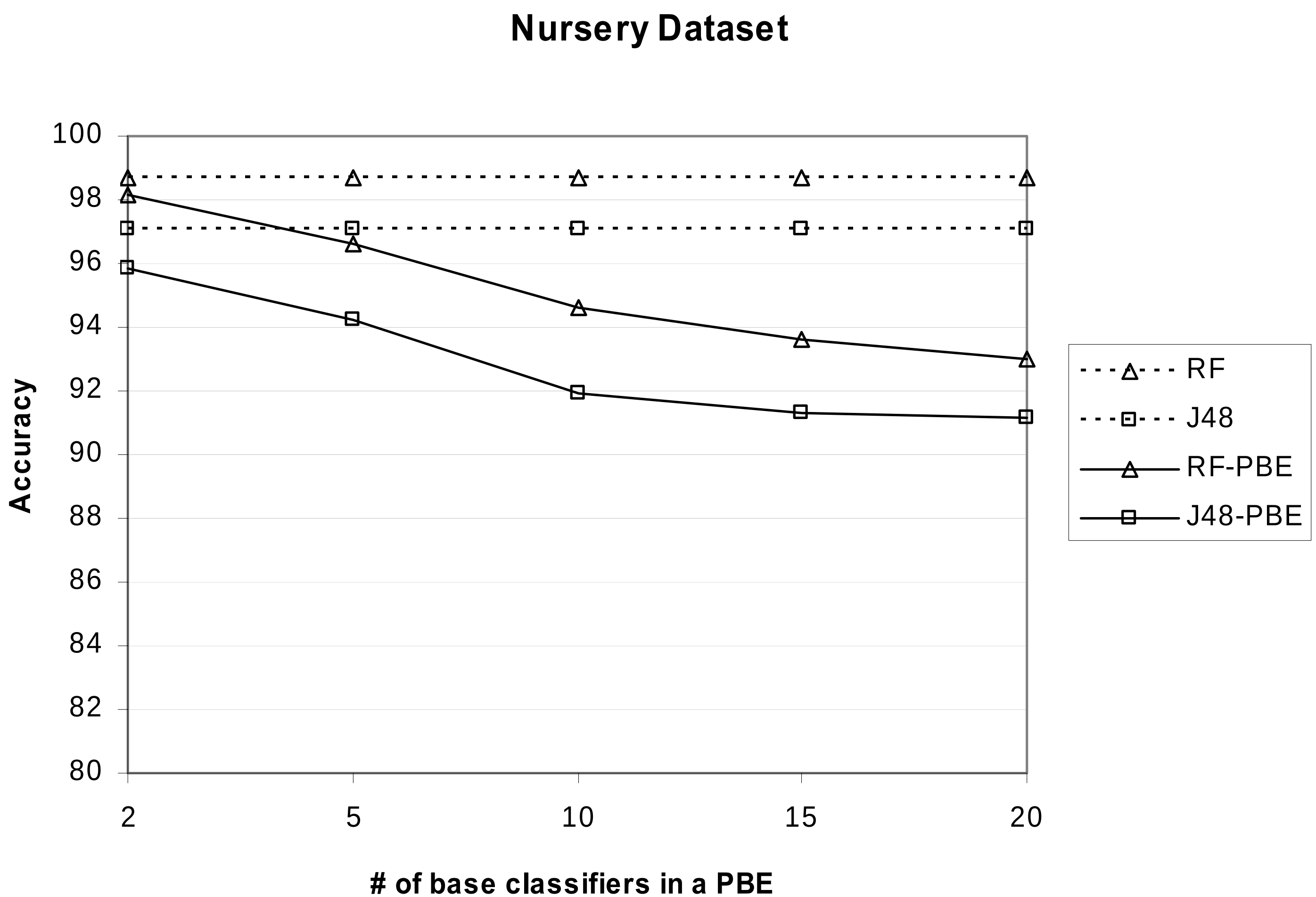
Accuracy of PBE

- 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



Accuracy of PBE



Accuracy of PBE

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