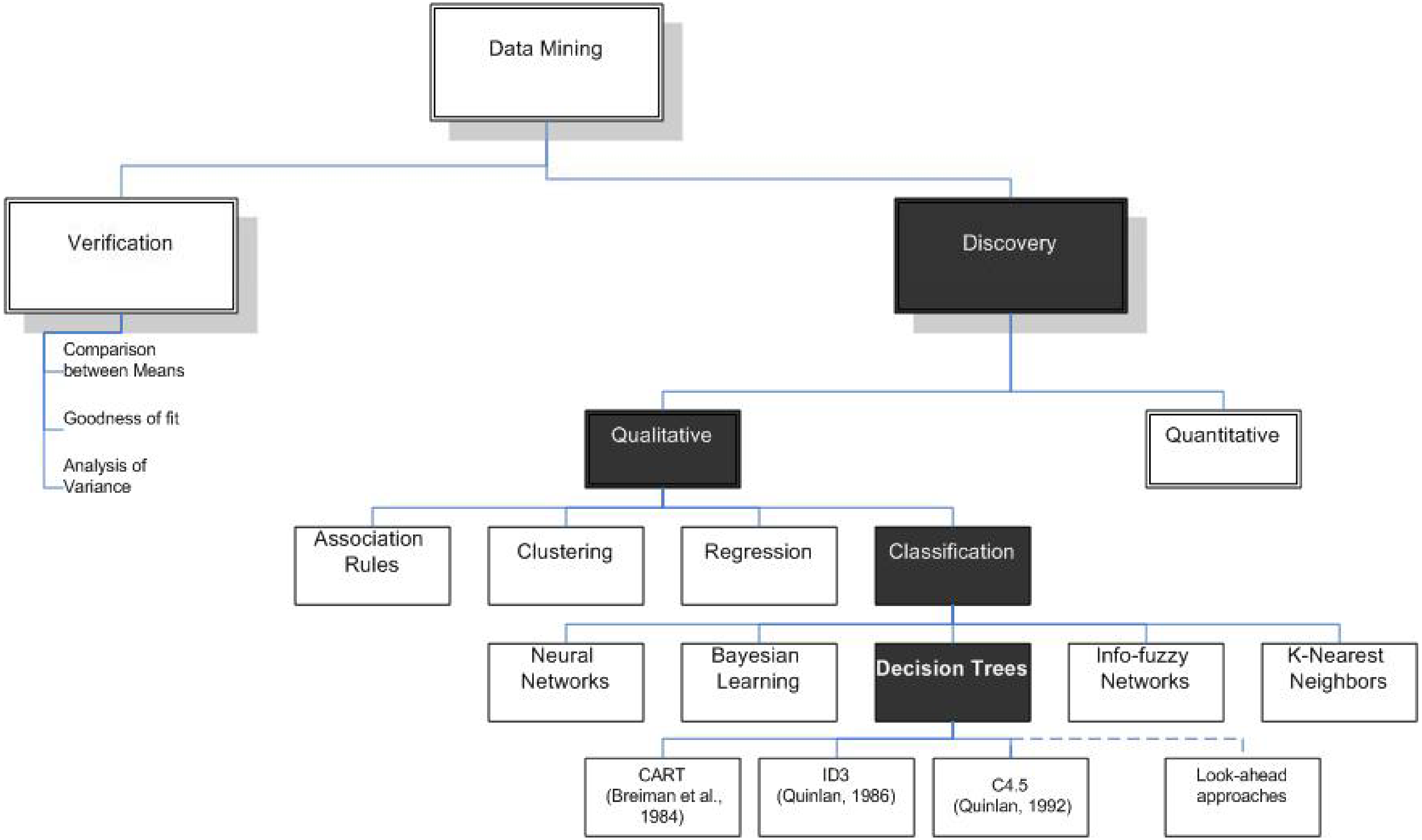
Efficient Construction of Decision Trees by the Dual Information Distance Method

Irad Ben-Gal
Joint work with Alexandra Dana, Niv Shkolnik and Gonen Singer
Dept. of Industrial Engineering
Tel Aviv University

Outline

1. **Introduction & Motivation**
2. Proposed Partitions Approach
3. Example
4. Results
5. Mid-level solutions
6. Summary & Contribution

Introduction



Based on Maimon & Rokach (2005)

Introduction

Classification (Supervised Learning)

"In classification, there is a target categorical variable, which is partitioned into predetermined classes or categories. The data mining model examine a large set of records, each record containing information on the target variable as well as a set of input or predictor variables". (Larose, 2005).

Introduction

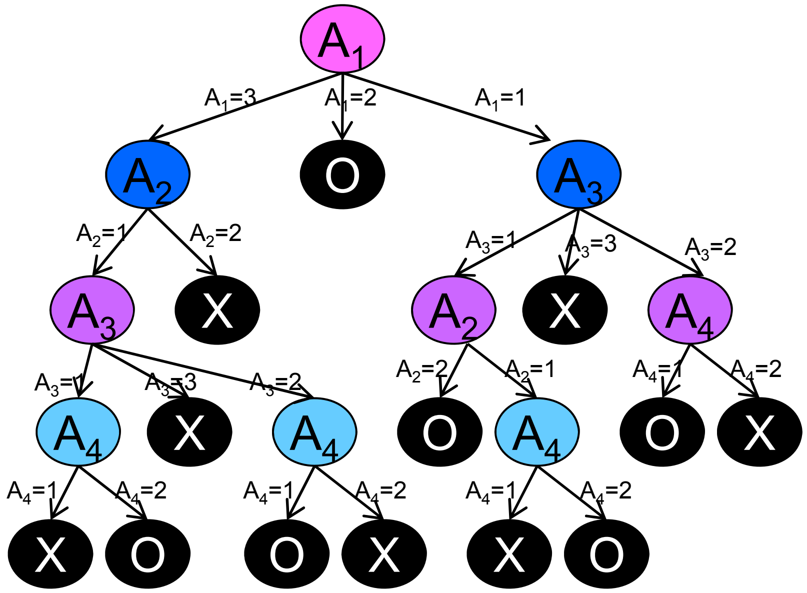
Construction of Decision Tree

- Classification variables
  - Class/Target variable Y
  - Attributes set $\{A_1,A_2,\ldots,A_n\}$
- The tree partitions Y by selecting attributes A_i, aiming that each leaf will contain a single class of Y
- Performance measures
  - Minimizing the classification error-rates
  - Minimizing the average depth of the tree
  - Minimizing the number of nodes/leaves

Introduction

#	A1	A2	A3	A4	Y
1	1	1	1	1	X
2	1	1	1	2	O
3	1	1	2	1	O
4	1	1	2	2	X
5	1	1	3	2	X
6	1	2	1	1	O
7	1	2	1	2	O
8	1	2	3	2	X
9	2	1	1	1	O
10	2	1	1	2	O
11	2	1	2	1	O
12	2	2	1	1	O
13	2	2	1	2	O
14	2	2	2	1	O
15	2	2	2	2	O
16	3	1	1	2	X
17	3	1	2	1	X
18	3	1	2	2	O
19	3	1	3	1	X
20	3	1	3	2	X
21	3	2	2	1	X
22	3	2	2	2	X
23	3	2	3	1	X
24	3	1	1	1	O
25	3	2	3	2	X

Example: ID3 (Lee & Olafsson, 2006)



ID3 Decision Tree example

- 4 categorical attributes $A_1, \ldots, A_4$
- Each selection splits the set into two or more partitions

Introduction

	A1	A2	A3	A4	Y
1	1	1	1	1	X
2	1	1	1	2	O
3	1	1	2	1	O
4	1	1	2	2	X
5	1	1	3	2	X
6	1	2	1	1	O
7	1	2	1	2	O
8	1	2	3	2	X
9	2	1	1	1	O
10	2	1	1	2	O
11	2	1	2	1	O
12	2	2	1	1	O
13	2	2	1	2	O
14	2	2	2	1	O
15	2	2	2	2	O
16	3	1	1	2	X
17	3	1	2	1	X
18	3	1	2	2	O
19	3	1	3	1	X
20	3	1	3	2	X
21	3	2	2	1	X
22	3	2	2	2	X
23	3	2	3	1	X
24	3	1	1	1	O
25	3	2	3	2	X

Partition by Y

$$\alpha_Y \equiv \left\langle \begin{array}{l} \{1,4,5,8,16,17,19,20,21,22,23,25\}, \\ \{2,3,6,7,9,10,11,12,13,14,15,18,24\} \end{array} \right\rangle$$

Introduction

	A1	A2	A3	A4	Y
1	1	1	1	1	X
2	1	1	1	2	O
3	1	1	2	1	O
4	1	1	2	2	X
5	1	1	3	2	X
6	1	2	1	1	O
7	1	2	1	2	O
8	1	2	3	2	X
9	2	1	1	1	O
10	2	1	1	2	O
11	2	1	2	1	O
12	2	2	1	1	O
13	2	2	1	2	O
14	2	2	2	1	O
15	2	2	2	2	O
16	3	1	1	2	X
17	3	1	2	1	X
18	3	1	2	2	O
19	3	1	3	1	X
20	3	1	3	2	X
21	3	2	2	1	X
22	3	2	2	2	X
23	3	2	3	1	X
24	3	1	1	1	O
25	3	2	3	2	X

Partition by A_1

$$\alpha_1 = \left\langle \begin{array}{l} \{1,2,3,4,5,6,7,8\}, \\ \{9,10,11,12,13,14,15\}, \\ \{16,17,18,19,20,21,22,23,24,25\} \end{array} \right\rangle$$

Introduction

	A1	A2	A3	A4	Y
1	1	1	1	1	X
2	1	1	1	2	O
3	1	1	2	1	O
4	1	1	2	2	X
5	1	1	3	2	X
6	1	2	1	1	O
7	1	2	1	2	O
8	1	2	3	2	X
9	2	1	1	1	O
10	2	1	1	2	O
11	2	1	2	1	O
12	2	2	1	1	O
13	2	2	1	2	O
14	2	2	2	1	O
15	2	2	2	2	O
16	3	1	1	2	X
17	3	1	2	1	X
18	3	1	2	2	O
19	3	1	3	1	X
20	3	1	3	2	X
21	3	2	2	1	X
22	3	2	2	2	X
23	3	2	3	1	X
24	3	1	1	1	O
25	3	2	3	2	X

Partition by $A_1 \vee A_2$

$$\alpha_{12} = \left\langle \{1,2,3,4,5\},\{6,7,8\}, \{9,10,11\},\{12,13,14,15\}, \{16,17,18,19,20,24\},\{21,22,23,25\} \right\rangle$$

Introduction

Types of Decision Tree Algorithms

Optimal Decision Trees

- ❑ Consider all possible splits (all combinations)
- ❑ Construction is NP-hard (Hackock et al., 1996)

Heuristic Trees

- ❑ Greedy trees (ID3, C4.5, CART)
  - At each step consider only the next split
- ❑ Look-ahead trees
  - consider up-coming splits (usually 2-steps ahead)
  - Computationally "pricy": $O(mn^K)$ for n variables; m records, and a K-steps look-ahead procedure

Introduction

Types of Decision Tree Algorithms

Heuristic Trees

Can we do better than greedy selection without extensive computations?

Maybe: e.g. by using the Dual Information Distance (DID) approach on a ‘partitions graph’

Introduction

ID3 (Quinlan, 1986) & C4.5 (Quinlan, 1993)

- Recursively split each node until no splits are possible
- ID3 Splitting Criteria: (highest) Information Gain

$$\text{Information Gain}(Y;A_i) = H(Y) - H(Y \mid A_i)$$

- C4.5 Splitting Criteria: (highest) Gain Ratio

$$\text{Gain Ratio}(Y;A_i) = \frac{\text{Information Gain}(Y;A_i)}{\text{Enropy}(A_i)}$$

- Max information gain/Reduction of entropy (uncertainty)!
- Very popular & produces good results in practice problems (Goodman & Smyth, 1988; Murthy & Salzberg, 1995)

Outline

1. Introduction & Motivation
2. **IT Partitions Approach**
3. Example
4. Results
5. Mid-level solutions
6. Summary & Contribution

Information Theory

Chain Rule of Mutual Information (Shannon, 48)

For two attributes, A_1, A_2 :

$$I(A_1, A_2; Y) = I(A_1; Y) + I(A_2; Y | A_1) =$$

$$H(A_1) + H(A_2 | A_1) - H(A_1 | Y) - H(A_2 | A_1, Y)$$

In General :

$$I(A_1, A_2, \ldots, A_n; Y) = \sum_i I(A_i; Y | A_{i-1}, \ldots, A_1) =$$

$$\sum_i \underbrace{H(A_i | A_{i-1}, \ldots, A_1)}_{\text{Max Orthogonality (DOE)}} - \sum_i \underbrace{H(A_i | A_{i-1}, \ldots, A_1, Y)}_{\text{Minimize Entropy / Remaining uncertainty (DM)}}$$

Information Theory

Chain Rule of Mutual Information (Shannon, 48)

In General:

$$I(A_1, A_2, \ldots, A_n; Y) = \sum_i I(A_i; Y \mid A_{i-1}, \ldots, A_1) =$$

$$\underbrace{\sum_i H(A_i \mid A_{i-1}, \ldots, A_1)}_{\text{Max Orthogonality (DOE)}} - \underbrace{\sum_i H(A_i \mid A_{i-1}, \ldots, A_1, Y)}_{\text{Minimize Entropy / Remaining uncertainty (DM)}}$$

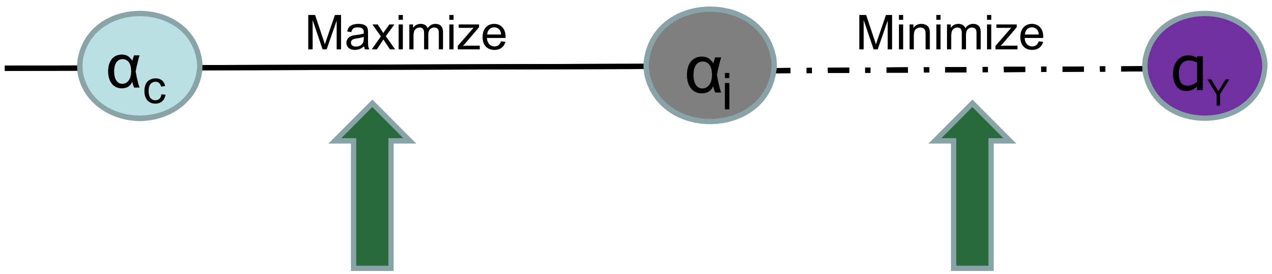
Use a Dual Information Distance approach, wrt:

1. Current state / partition $\alpha_c = A_{i-1} \vee A_{i-2}, \ldots, A_1$
2. Target variable

The DID Partition Approach

Orthogonality and Information Gain

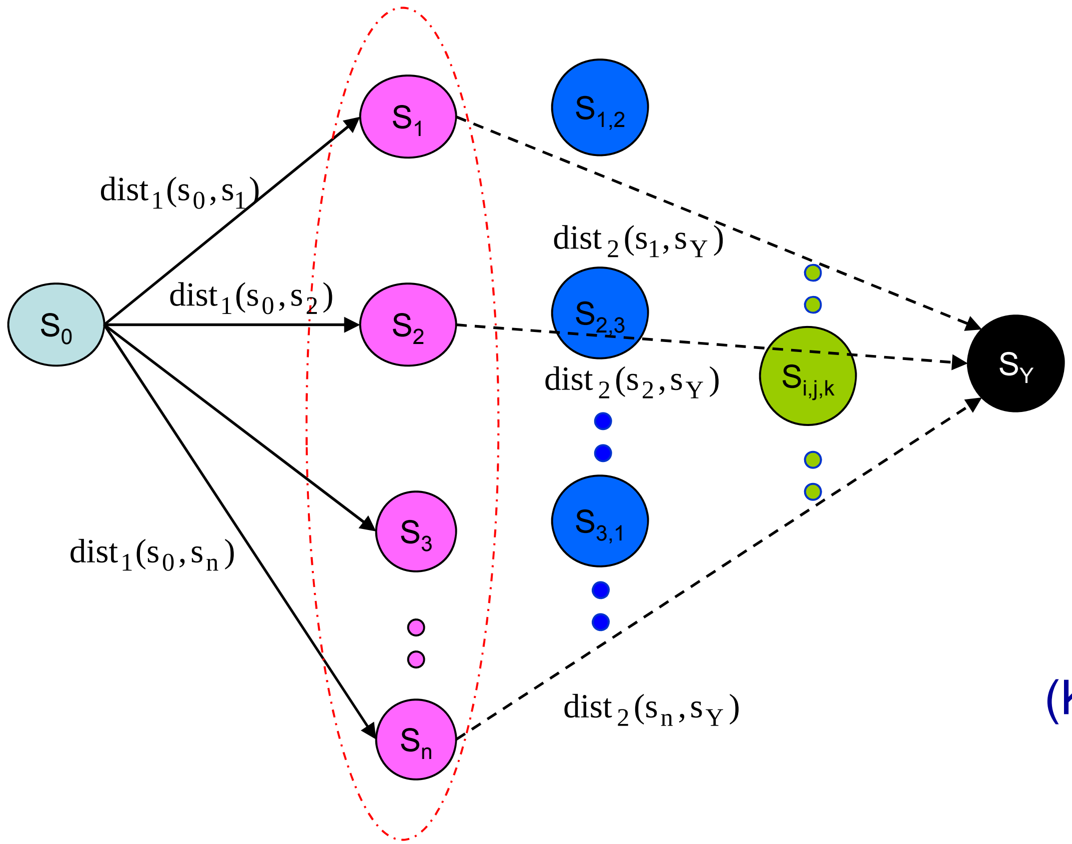
- ❑ **When selecting attribute A_i examine the resulting partition with respect to dual Inf. distance:**
- ✓ **Current partition (current tree state)**
- ✓ **Target/class partition**



Orthogonality
(DOE approach)

Remaining Uncertainty
(DM approach)

LRTA* Algorithm



(Korf, 1990)

Applying LRTA* concepts to Decision-Tree Construction

State s_i ↔ partition α_i
Neighbors of state s_i ↔ Refinement of partition α_i (add an attribute)
Distance *d*(s_i,s_j) ↔ A distance metric defined over partitions

The DID Approach

Rokhlin (60's) distance measure

$$Rokhlin(\alpha, \beta) = H(\alpha|\beta) + H(\beta|\alpha)$$

- **This distance follows the required properties of a metric:**

$$d(\alpha, \beta) \geq 0$$

$$d(\alpha, \alpha) = 0$$

$$d(\alpha, \beta) \leq d(\alpha, \gamma) + d(\gamma, \beta), \forall \alpha, \beta, \gamma \epsilon \chi$$

- **Relation between mutual information and Rokhlin metric**

$$d(\alpha, \beta) = H(\alpha, \beta) - I(\alpha, \beta)$$

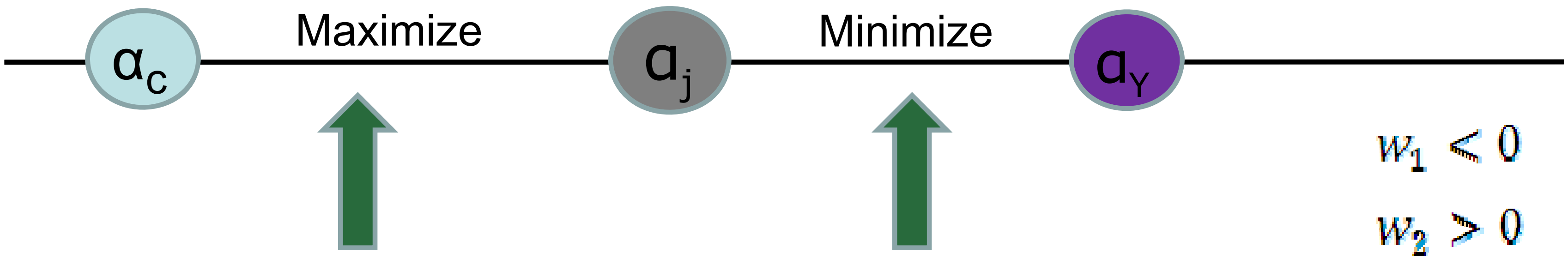
The DID Approach

A **general objective function**

$$\min_{\alpha_j}\{w_1 d_1(\alpha_c, \alpha_j) + w_2 d_2(\alpha_j, \alpha_Y)$$

d_1 denotes the orthogonality measure distance between the current partition and the next chosen partition

d_2 denotes the information (or remaining uncertainty) distance between the chosen partition and the class partition



$w_1 < 0$

$w_2 > 0$

Orthogonality
(DOE approach)

Remaining Uncertainty
(DM approach)

Korf, 1990

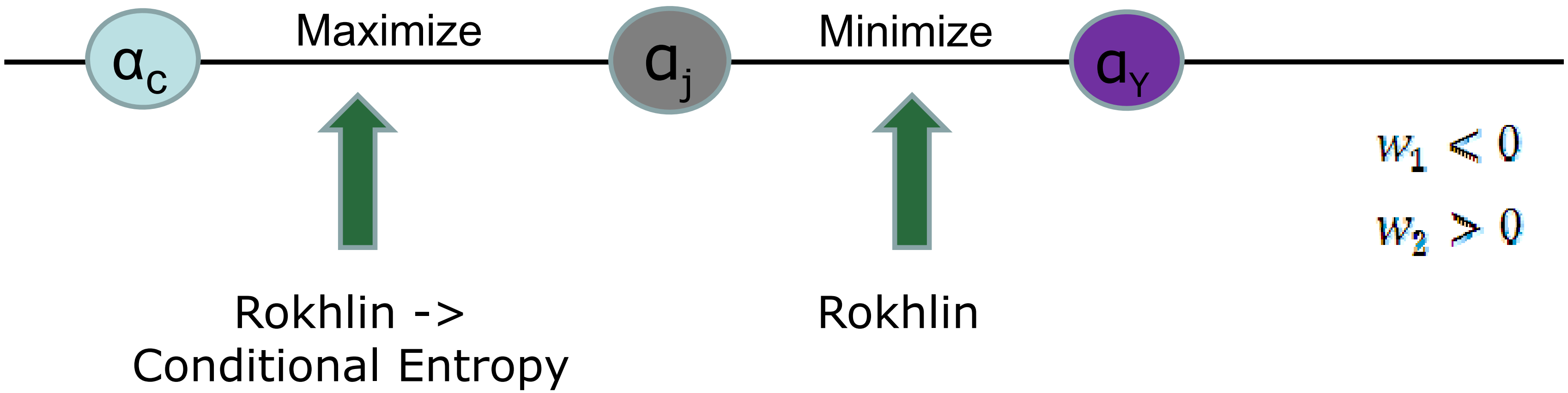
The DID Approach

A **general objective function**

$$\min_{\alpha_j}\{w_1 d_1(\alpha_c, \alpha_j) + w_2 d_2(\alpha_j, \alpha_Y)$$

d_1 denotes the orthogonality measure distance between the current partition and the next chosen partition

d_2 denotes the information (or remaining uncertainty) distance between the chosen partition and the class partition



$w_1 < 0$

$w_2 > 0$

Korf, 1990

The DID Approach

Orthogonality between restricted partitions

- Let us consider the following example:
- For A_1 partition A_3 is most orthogonal

$$H(A_3 \mid A_1) = 0.63 \quad H(A_2 \mid A_1) = 1.24$$

A_1	A_2	**A_3**	*Y*
1	1	1	2
1	1	2	2
1	1	2	2
1	2	2	3
1	2	2	3
1	2	2	3
2	2	1	1
2	2	1	1
2	2	1	1
2	2	2	1
2	2	2	1
2	2	2	1

The DID Approach

Orthogonality between restricted partitions

- Let us consider the following example:
- A_1 and A_3 are most orthogonal, however they **do not classify *Y***

A_1	A_2	**A_3**	*Y*
1	1	1	2
1	1	2	**2**
1	1	2	**2**
1	2	2	**3**
1	2	2	**3**
1	2	2	**3**
2	2	1	1
2	2	1	1
2	2	1	1
2	2	2	1
2	2	2	1
2	2	2	1

The DID Approach

Orthogonality between restricted partitions

- Let us consider the following example:
- A_1 and A_3 are most orthogonal, however they do not classify Y
- Focusing on the orthogonality between the restricted partitions:

  For A_1 =1: A_2 is most orthogonal

  For A_1 =2: A_3 is most orthogonal but this sub-set is already classified by A_1

A_1	A_2	A_3	Y
1	1	1	2
1	1	2	2
1	1	2	2
1	2	2	3
1	2	2	3
1	2	2	3
2	2	1	1
2	2	1	1
2	2	1	1
2	2	2	1
2	2	2	1
2	2	2	1

The DID Approach

Orthogonality between restricted partitions

- Let us consider the following example:
- A_1 and A_3 are most orthogonal, however they do not classify *Y*
- Focusing on the orthogonality between the restricted partitions:

  For A_1 =1: A_2 is most orthogonal

  For A_1 =2: A_3 is most orthogonal but this sub-set is already classified by A_1
- Full classification of *Y* is achieved
- The DM "Preprocess Approach" is not always helpful!

A_1	A_2	A_3	*Y*
1	1	1	2
1	1	2	2
1	1	2	2
1	2	2	3
1	2	2	3
1	2	2	3
2	2	1	1
2	2	1	1
2	2	1	1
2	2	2	1
2	2	2	1
2	2	2	1

The DID Approach

Rokhlin remainig uncertainty measure

- The Rokhlin distance measure takes into account two terms:

  $H(Y|A_i)$ $H(A_i|Y)$

A_1	A_2	Y
1	1	X
2	1	X
3	1	X
4	2	O
5	2	O
6	2	O

The DID Approach

Rokhlin distance measure

- The Rokhlin distance measure takes into account two terms:

$H(Y|A_i)$ $\quad$ $H(A_i|Y)$

$H(Y|A_1) = 0$

$H(Y|A_2) = 0$

A_1	A_2	Y
1	1	X
2	1	X
3	1	X
4	2	O
5	2	O
6	2	O

The DID Approach

Rokhlin distance measure

- The Rokhlin distance measure takes into account two terms:

  $H(Y|A_i)$ $H(A_i|Y)$

  $H(Y|A_1) = 0$

  $H(Y|A_2) = 0$

  $H(A_1|Y) = 1.58$

  $H(A_2|Y) = 0$

A_1	A_2	Y
1	1	X
2	1	X
3	1	X
4	2	O
5	2	O
6	2	O

$\{\alpha_1^1, \alpha_1^2, \ldots, \alpha_1^6\}$,

$\{\alpha_2^1, \alpha_2^2\}$

- Prefer A_2 over A_1
- Partition the dataset "as little" as required

The DID Approach

Notation

- α_0 - The "initial partition": $\{\{X\},\{\emptyset\}\}$
- α_Y - The "class/target partition"
- α_c - current partition (current state of the tree)
- α_i - the partition that results from the selection of attribute *i*
- α_i^j - sub-partition of α_i where the levels of attribute *i* are equal to *j* $\alpha_i = \{\alpha_i^j \mid \alpha_i^j \in \alpha_i, \alpha_i^j \cap \alpha_i^k = \emptyset, \bigcup_j \alpha_i^j = \alpha_i\}$
- Selecting an attribute in a node may results in a refinement of the current partition, i.e., $\alpha_i \vee \alpha_j$ is a refinement of α_i

$$\alpha_i \vee \alpha_j = \{\alpha_i^l \cap \alpha_j^k \mid l = 1,2,\dots,|\alpha_i|, k = 1,2,\dots,|\alpha_j|\}$$

The DID Approach

Information and Remaining Uncertainty

- Look for a partition that results in maximum information (minimum classification uncertainty)
- Using **Entropy**: choosing the partition i which gives minimum $H(\alpha_Y | \alpha_i)$
- Using **Rokhlin**: choosing the partition i which gives minimum $R(\alpha_Y, \alpha_i) = H(\alpha_Y | \alpha_i) + H(\alpha_i | \alpha_Y)$

Minimizing the classification uncertainty

staying "as close as possible" to α_Y avoiding unnecessary refinement

The Proposed DID Algorithm

Algorithm (DID)

Given i) set of weights w_1, w_2; ii) two distance metrics denoted by d_1 and d_2;
iii) *attributes partitions* $\alpha_1, \alpha_2, \ldots, \alpha_n$; and iv) a class partition α_Y

Do:

Init current partition $\alpha_c \leftarrow \alpha_0$

Init $E = \{\emptyset\}, F = \{1,2,\ldots,n\}$ *<groups of the "used" and the "unused" attributes >*

For each sub-partition $\alpha_c^i \epsilon \alpha_c$ *such that i)* $|\alpha_c^i| > 1$; *ii)* $\alpha_Y|_{\alpha_c^i}$ *is not yet classified; and iii) F is not empty*

start the **Search** *procedure (for the sub-partition* α_c^i*)*

The Proposed DID Algorithm

Algorithm (DID)

Given i) set of weights w_1, w_2; ii) two distance metrics denoted by d_1 and d_2;

iii) *attributes partitions* $\alpha_1, \alpha_2, \dots, \alpha_n$; and iv) a class partition α_Y

Do:

Init current partition $\alpha_c \leftarrow \alpha_0$

Init $E = \{\emptyset\}, F = \{1,2,\dots,n\}$ *<groups of the "used" and the "unused" attributes >*

For each sub-partition $\alpha_c^i \epsilon \alpha_c$ *such that i)* $|\alpha_c^i| > 1$; *ii)* $\alpha_Y|_{\alpha_c^i}$ *is not yet classified; and iii) F is not empty*

start the **Search** *procedure (for the sub-partition* α_c^i*)*

The Proposed DID Algorithm

Algorithm (DID)

Given i) set of weights w_1, w_2; ii) two distance metrics denoted by d_1 and d_2;

iii) *attributes partitions* $\alpha_1, \alpha_2, \ldots, \alpha_n$; and iv) a class partition α_Y

Do:

Init current partition $\alpha_c \leftarrow \alpha_0$

Init $E = \{\emptyset\}, F = \{1,2,\ldots,n\}$ *<groups of the "used" and the "unused" attributes >*

For each sub-partition $\alpha_c^i \epsilon \alpha_c$ *such that i)* $|\alpha_c^i| > 1$; *ii)* $\alpha_Y|_{\alpha_c^i}$ *is not yet classified; and iii) F is not empty*

start the **Search** *procedure (for the sub-partition* α_c^i*)*

The Proposed DID Algorithm

Function Search. *Given set* α_c^i *and the attributes partitions* $\alpha_1, \alpha_2, \ldots, \alpha_n$; E; F;

1. *Init current partition* α_c *by* α_c^i; *init* $E_c \leftarrow E$; $F_c \leftarrow F$
2. *Normalize probabilities of the elements of* α_c^i.
3. *Create local class partition* $\alpha_Y|_{\alpha_c}$.
4. *Generate neighborhood partitions*:

   $N(\alpha_c) = \{\alpha_j|_{\alpha_c}\}, j \epsilon F_c, F_c = \{j : A_j \text{ not selected by the algorithm yet}\}$
5. *Normalize probabilities of neighbors and of the class partition;*
6. *Obtain distance measures by* $d_1(\alpha_c, \alpha_j|_{\alpha_c})$ *and* $d_2(\alpha_j|_{\alpha_c}, \alpha_Y|_{\alpha_c}), j \epsilon F_c$
7. *Choose next partition*: The next partition is selected as follows (ties are resolved arbitrary):

   $$\alpha_{next} \leftarrow arg\,min_{\alpha_j \in N(\alpha_c)} \{w_1 d_1(\alpha_c, \alpha_j|_{\alpha_c}) + w_2 d_2(\alpha_j|_{\alpha_c}, \alpha_Y|_{\alpha_c})\}$$
8. *Update* E_c *and* F_c *(move j from* F_c *to* E_c*)*
9. *Move to next partition:* $\alpha_c \leftarrow \alpha_{next}$
10. *For each sub-partition* $\alpha_c^i \epsilon \alpha_c$ *such that i)* $|\alpha_c^i| > 1$; *ii)* $\alpha_Y|_{\alpha_c^i}$ *is not yet classified; and iii)* F_c *is not empty*

    start the **Search** *procedure (for sub-partition* α_c^i, F_c, E_c*)*

If $\alpha_Y|_{\alpha_c}$ *is classified, return.*

If $|\alpha_c| = 1$ *classify according to the instance's class value, return.*

If $F_c = \{\emptyset\}$ *classify according to the most common value of the class attribute, return.*

The Proposed DID Algorithm

1. *Init current partition* α_c *by* α_c^i; *init* $E_c \leftarrow E$; $F_c \leftarrow F$
2. *Normalize probabilities of the elements of* α_c^i.
3. *Create local class partition* $\alpha_Y|_{\alpha_c}$.
4. *Generate neighborhood partitions:*

   $$N(\alpha_c) = \{\alpha_j|_{\alpha_c}\}, j \epsilon F_c, F_c = \{j : A_j \text{ not selected by the algorithm yet}\}$$

5. *Normalize probabilities of neighbors and of the class partition;*

resolved arbitrary):

$$\alpha_{next} \leftarrow arg\,min_{\alpha_j \in N(\alpha_c)} \{w_1 d_1(\alpha_c, \alpha_j|_{\alpha_c}) + w_2 d_2(\alpha_j|_{\alpha_c}, \alpha_Y|_{\alpha_c})\}$$

8. *Update* E_c *and* F_c *(move j from* F_c *to* E_c*)*
9. *Move to next partition:* $\alpha_c \leftarrow \alpha_{next}$
10. *For each sub-partition* $\alpha_c^i \epsilon \alpha_c$ *such that i)* $|\alpha_c^i| > 1$; *ii)* $\alpha_Y|_{\alpha_c^i}$ *is not yet classified; and iii)* F_c *is not empty*

    start the **Search** *procedure (for sub-partition* α_c^i, F_c, E_c*)*

If $\alpha_Y|_{\alpha_c}$ *is classified, return.*

If $|\alpha_c| = 1$ *classify according to the instance's class value, return.*

If $F_c = \{\emptyset\}$ *classify according to the most common value of the class attribute, return.*

The Proposed DID Algorithm

Function Search. *Given set* α_c^i *and the attributes partitions* $\alpha_1, \alpha_2, \ldots, \alpha_n$; E; F;

1. *Init current partition* α_c *by* α_c^i; *init* $E_c \leftarrow E$; $F_c \leftarrow F$
2. *Normalize probabilities of the elements of* α_c^i.
3. *Create local class partition* $\alpha_Y|_{\alpha_c}$.
4. *Generate neighborhood partitions*:

   $N(\alpha_c) = \{\alpha_j|_{\alpha_c}\}, j \epsilon F_c, F_c = \{j: A_j$ not selected by the algorithm yet$\}$

6. *Obtain distance measures by* $d_1(\alpha_c, \alpha_j|_{\alpha_c})$ *and* $d_2(\alpha_j|_{\alpha_c}, \alpha_Y|_{\alpha_c}), j \epsilon F_c$
7. *Choose next partition*: The next partition is selected as follows (ties are resolved arbitrary):

   $$\alpha_{next} \Leftarrow arg\,min_{\alpha_j \in N(\alpha_c)} \{w_1 d_1(\alpha_c, \alpha_j|_{\alpha_c}) + w_2 d_2(\alpha_j|_{\alpha_c}, \alpha_Y|_{\alpha_c})\}$$

8. *Update* E_c *and* F_c *(move j from* F_c *to* E_c*)*
9. *Move to next partition:* $\alpha_c \Leftarrow \alpha_{next}$
10. *For each sub-partition* $\alpha_c^i \epsilon \alpha_c$ *such that i)* $|\alpha_c^i| > 1$; *ii)* $\alpha_Y|_{\alpha_c^i}$ *is not yet classified; and iii)* F_c *is not empty*

    start the **Search** *procedure (for sub-partition* α_c^i, F_c, E_c*)*

If $\alpha_Y|_{\alpha_c}$ *is classified, return.*

If $|\alpha_c| = 1$ *classify according to the instance's class value, return.*

If $F_c = \{\emptyset\}$ *classify according to the most common value of the class attribute, return.*

The Proposed DID Algorithm

Function Search. *Given set α_c^i and the attributes partitions $\alpha_1, \alpha_2, \dots, \alpha_n$; E; F;*

1. *Init current partition α_c by α_c^i; init $E_c \leftarrow E$; $F_c \leftarrow F$*
2. *Normalize probabilities of the elements of α_c^i.*
3. *Create local class partition $\alpha_Y|_{\alpha_c}$.*
4. *Generate neighborhood partitions:*

   $N(\alpha_c) = \{\alpha_j|_{\alpha_c}\}, j \epsilon F_c, F_c = \{j: A_j \text{ not selected by the algorithm yet}\}$
5. *Normalize probabilities of neighbors and of the class partition;*
6. *Obtain distance measures by $d_1(\alpha_c, \alpha_j|_{\alpha_c})$ and $d_2(\alpha_j|_{\alpha_c}, \alpha_Y|_{\alpha_c}), j \epsilon F_c$*
7. *Choose next partition*: The next partition is selected as follows (ties are resolved arbitrary):

)}
8. *Update E_c and F_c (move j from F_c to E_c)*
9. *Move to next partition: $\alpha_c \leftarrow \alpha_{next}$*
10. *For each sub-partition $\alpha_c^i \epsilon \alpha_c$ such that i) $|\alpha_c^i| > 1$; ii) $\alpha_Y|_{\alpha_c^i}$ is not yet classified; and iii) F_c is not empty*

    start the **Search** *procedure (for sub-partition α_c^i, F_c, E_c)*

If $\alpha_Y|_{\alpha_c}$ is classified, return.

If $|\alpha_c| = 1$ classify according to the instance's class value, return.

If $F_c = \{\emptyset\}$ classify according to the most common value of the class attribute, return.

The Proposed DID Algorithm

Function Search. *Given set* α_c^i *and the attributes partitions* $\alpha_1, \alpha_2, \ldots, \alpha_n$; E; F;

1. *Init current partition* α_c *by* α_c^i; *init* $E_c \leftarrow E$; $F_c \leftarrow F$
2. *Normalize probabilities of the elements of* α_c^i.
3. *Create local class partition* $\alpha_Y|_{\alpha_c}$.
4. *Generate neighborhood partitions*:

   $\mathrm{N}(\alpha_c) = \{\alpha_j|_{\alpha_c}\}, j\epsilon F_c, F_c = \{j: \mathrm{A}_j \text{ not selected by the algorithm yet}\}$
5. *Normalize probabilities of neighbors and of the class partition;*
6. *Obtain distance measures by* $d_1(\alpha_c, \alpha_j|_{\alpha_c})$ *and* $d_2(\alpha_j|_{\alpha_c}, \alpha_Y|_{\alpha_c}), j\epsilon F_c$
7. *Choose next partition*: The next partition is selected as follows (ties are resolved arbitrary):

   $$\alpha_{next} \leftarrow arg\,min_{\alpha_j \in N(\alpha_c)} \{w_1 d_1(\alpha_c, \alpha_j|_{\alpha_c}) + w_2 d_2(\alpha_j|_{\alpha_c}, \alpha_Y|_{\alpha_c})\}$$

10. *For each sub-partition* $\alpha_c^i \epsilon \alpha_c$ *such that i)* $|\alpha_c^i| > 1$; *ii)* $\alpha_Y|_{\alpha_c^i}$ *is not yet classified; and iii)* F_c *is not empty*

    start the **Search** *procedure (for sub-partition* α_c^i, F_c, E_c*)*

If $\alpha_Y|_{\alpha_c}$ *is classified, return.*

If $|\alpha_c| = 1$ *classify according to the instance's class value, return.*

If $F_c = \{\emptyset\}$ *classify according to the most common value of the class attribute, return.*

Outline

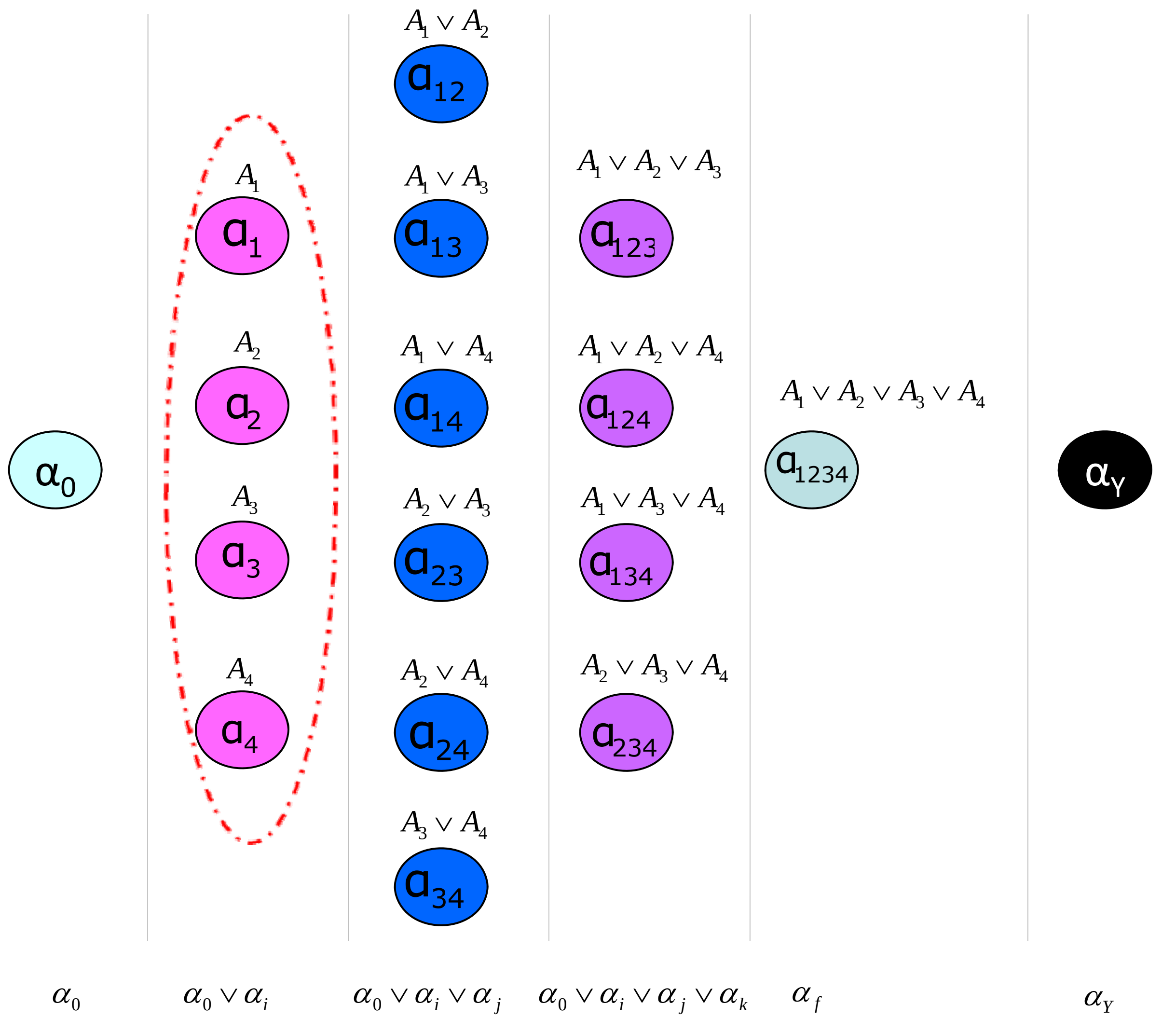
1. Introduction & Motivation
2. Proposed Partitions Approach
3. **Examples**
4. Results
5. Mid-level solutions
6. Summary & Contribution

Example

Training Data Set

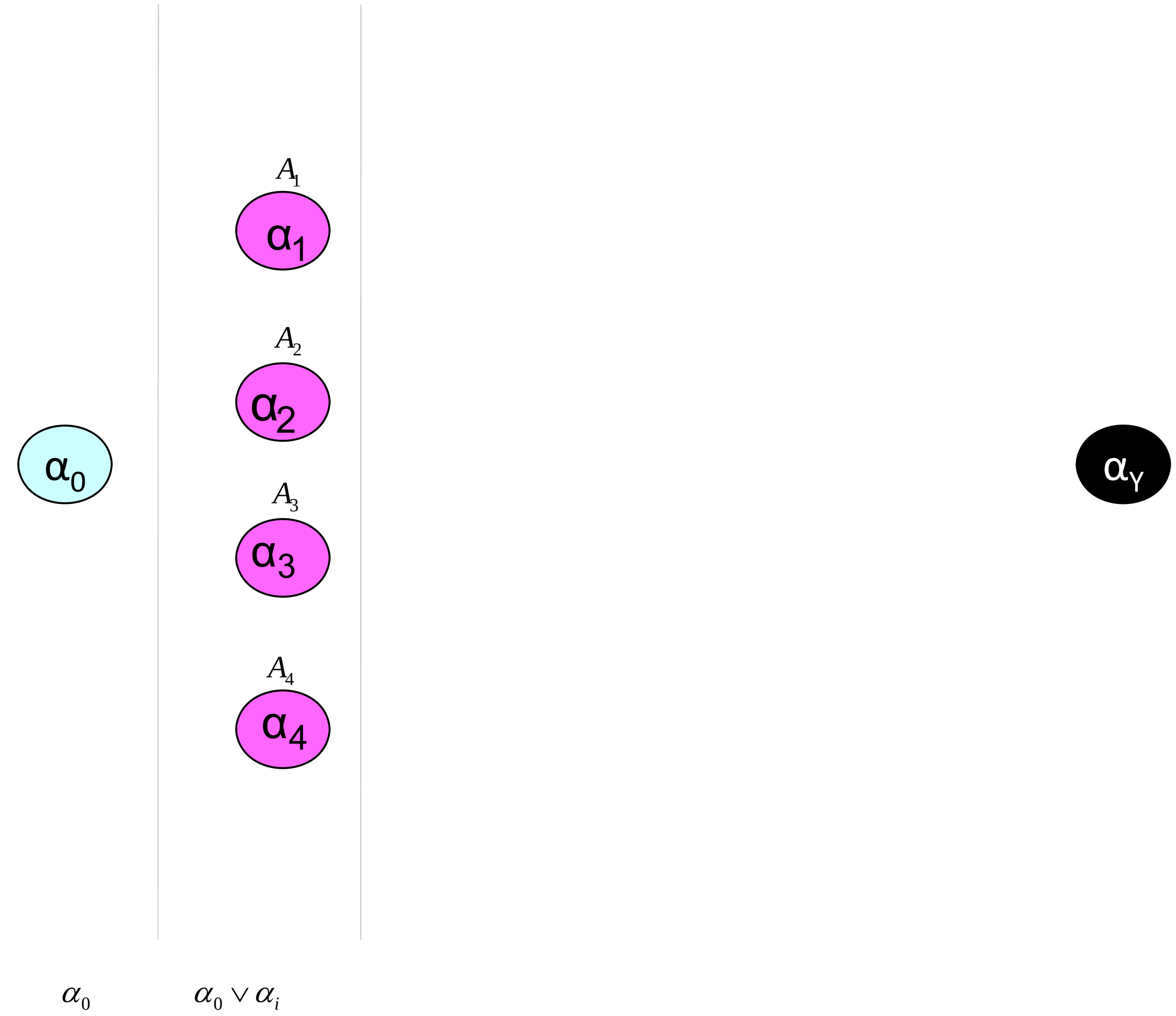
	Y	A1	A2	A3	A4
1	1	4	1	1	1
2	2	1	1	2	2
3	2	1	1	2	1
4	2	1	1	2	1
5	3	2	3	2	1
6	3	2	2	1	1
7	4	2	3	1	1
8	4	3	3	1	1
9	5	3	3	3	1
10	4	1	1	3	1
11	4	1	2	4	2
12	4	2	2	4	2

ID3 example



$A_1 \vee A_2$
a_{12}
A_1
a_1
$A_1 \vee A_3$
a_{13}
$A_1 \vee A_2 \vee A_3$
a_{123}
A_2
a_2
$A_1 \vee A_4$
a_{14}
$A_1 \vee A_2 \vee A_4$
a_{124}
$A_1 \vee A_2 \vee A_3 \vee A_4$
α_0
a_{1234}
α_Y
A_3
a_3
$A_2 \vee A_3$
a_{23}
$A_1 \vee A_3 \vee A_4$
a_{134}
A_4
a_4
$A_2 \vee A_4$
a_{24}
$A_2 \vee A_3 \vee A_4$
a_{234}
$A_3 \vee A_4$
a_{34}
α_0
$\alpha_0 \vee \alpha_i$
$\alpha_0 \vee \alpha_i \vee \alpha_j$
$\alpha_0 \vee \alpha_i \vee \alpha_j \vee \alpha_k$
α_f
α_Y

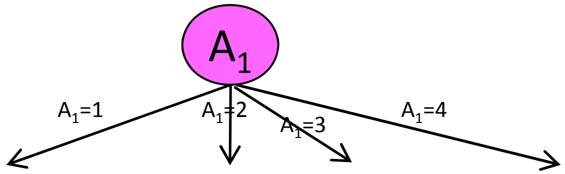
ID3 example



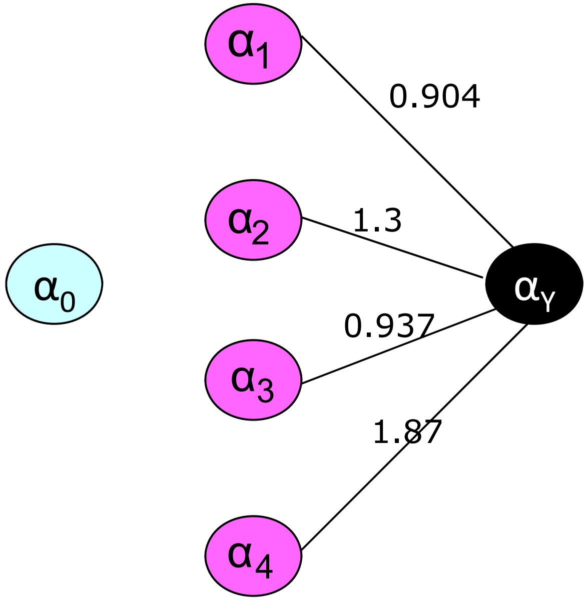
A_1
α_1
A_2
α_2
α_0
A_3
α_3
A_4
α_4
α_Y
α_0
$\alpha_0 \vee \alpha_i$

ID3 example

Classification Tree



Partitions Graph



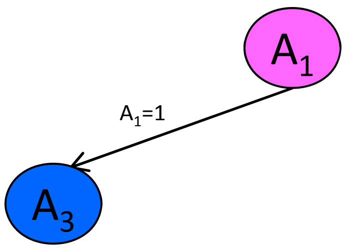
Results for :

$w_1 = 0; w_2 = +1;$

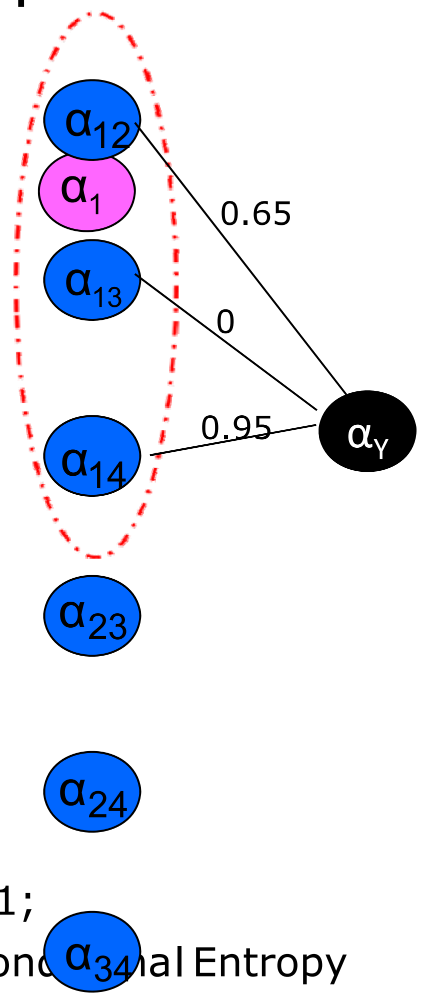
distance2 = Conditional Entropy

ID3 example

Classification Tree



Partitions Graph



Restricted to $A_1=1$ subset

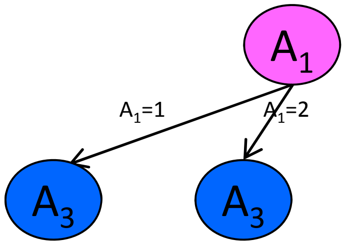
Results for :

$w_1 = 0; w_2 = +1;$

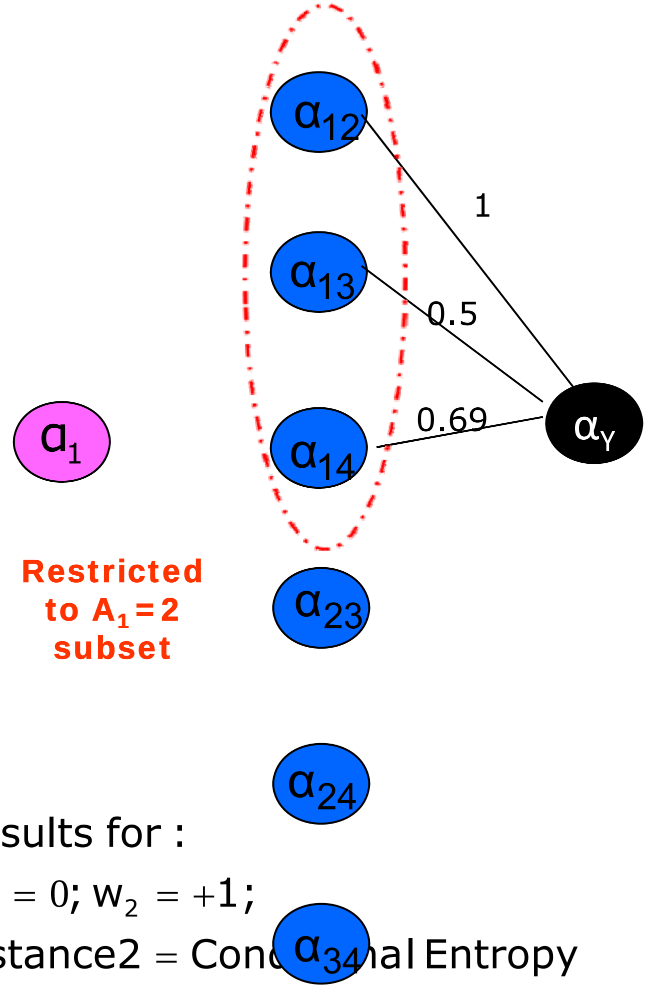
distance2 = Conditional Entropy

ID3 example

Classification Tree



Partitions Graph



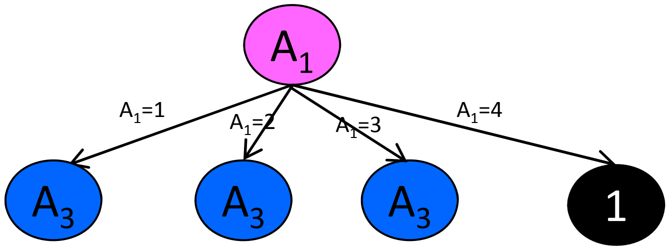
Results for :

$w_1 = 0$; $w_2 = +1$;

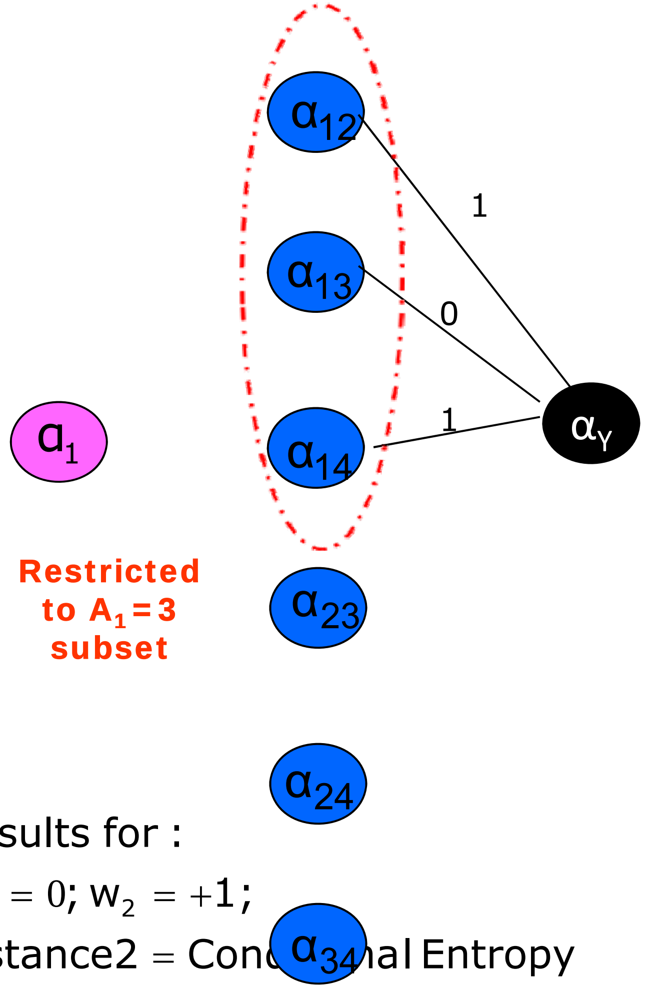
distance2 = Conditional Entropy

ID3 example

Classification Tree



Partitions Graph



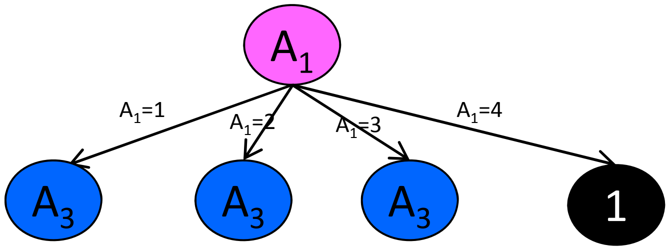
Results for :

$w_1 = 0$; $w_2 = +1$;

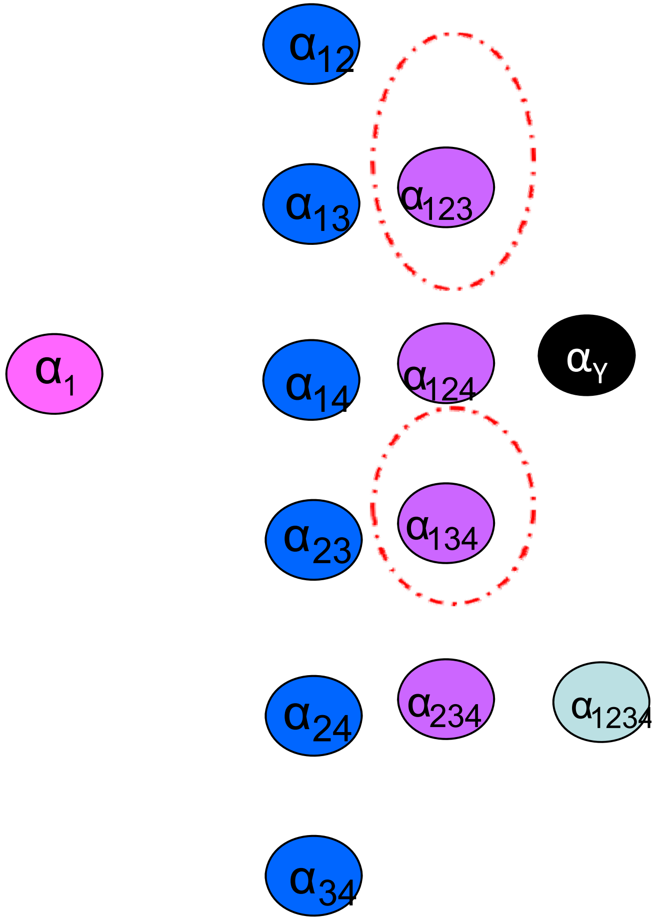
$distance2 =$ Conditional Entropy

ID3 example

Classification Tree

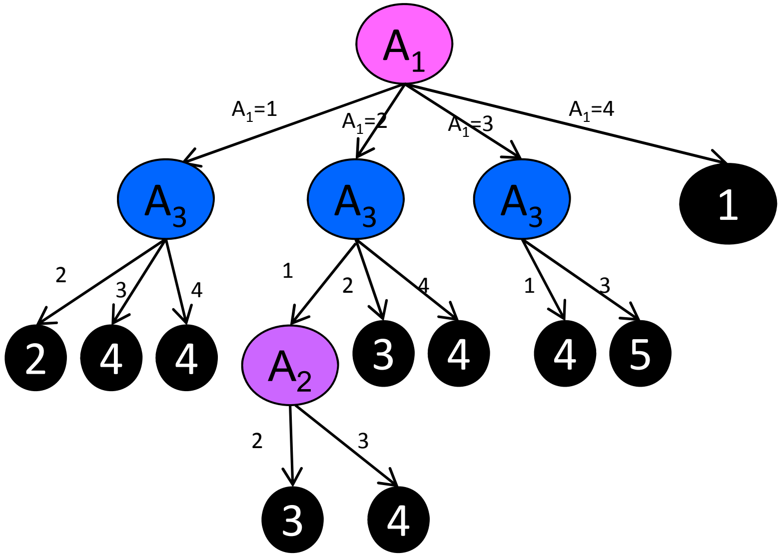


Partitions Graph



ID3 example

Classification Tree



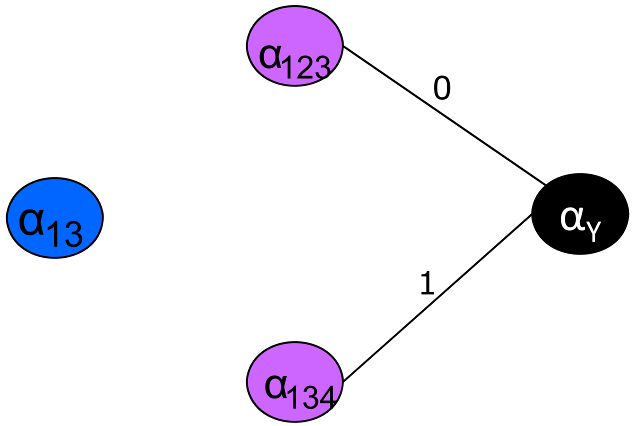
Average depth = 2.1

No. of decision = 5

No. of leaves = 10

Max steps = 3

Partitions Graph

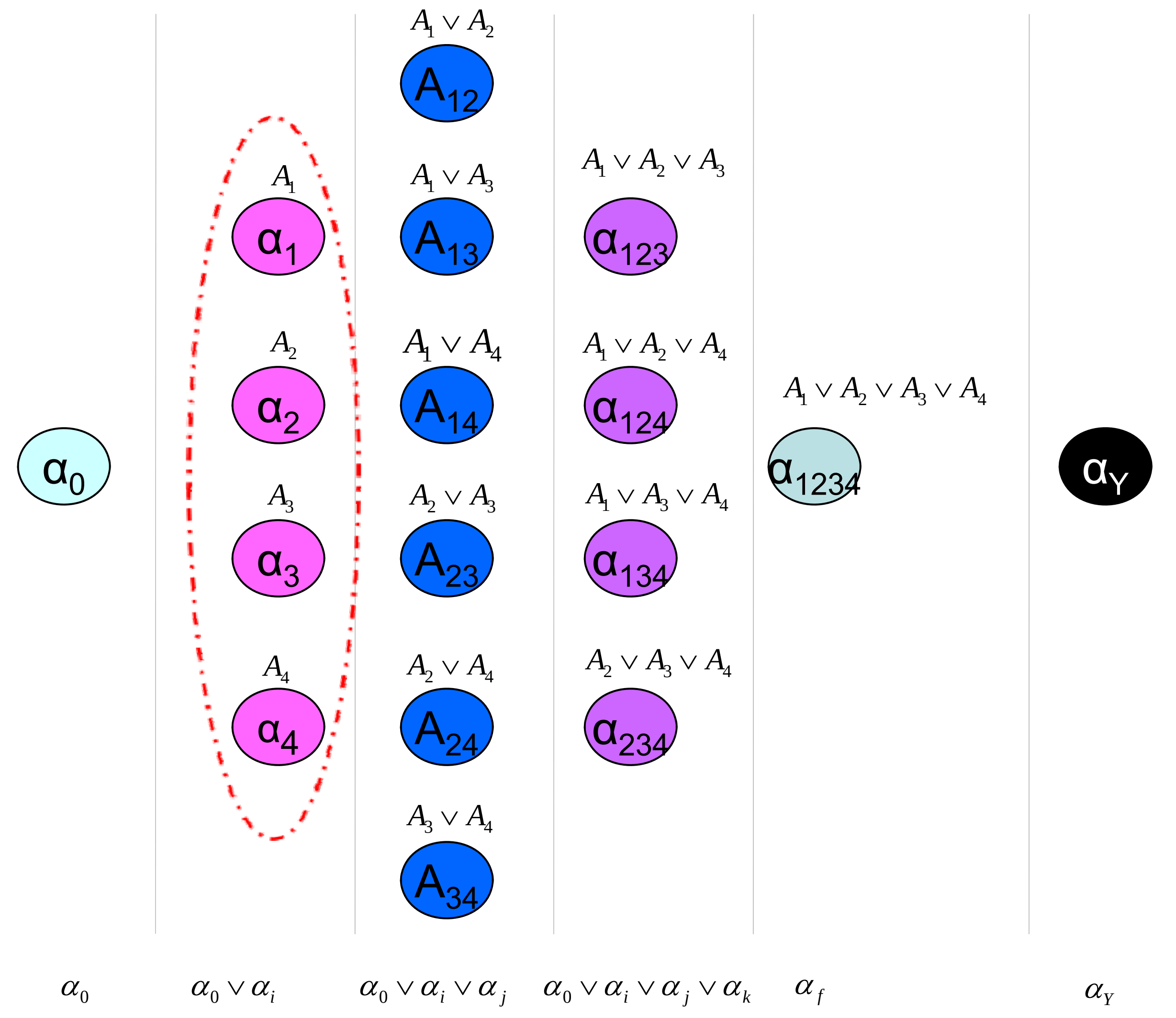


Results for :

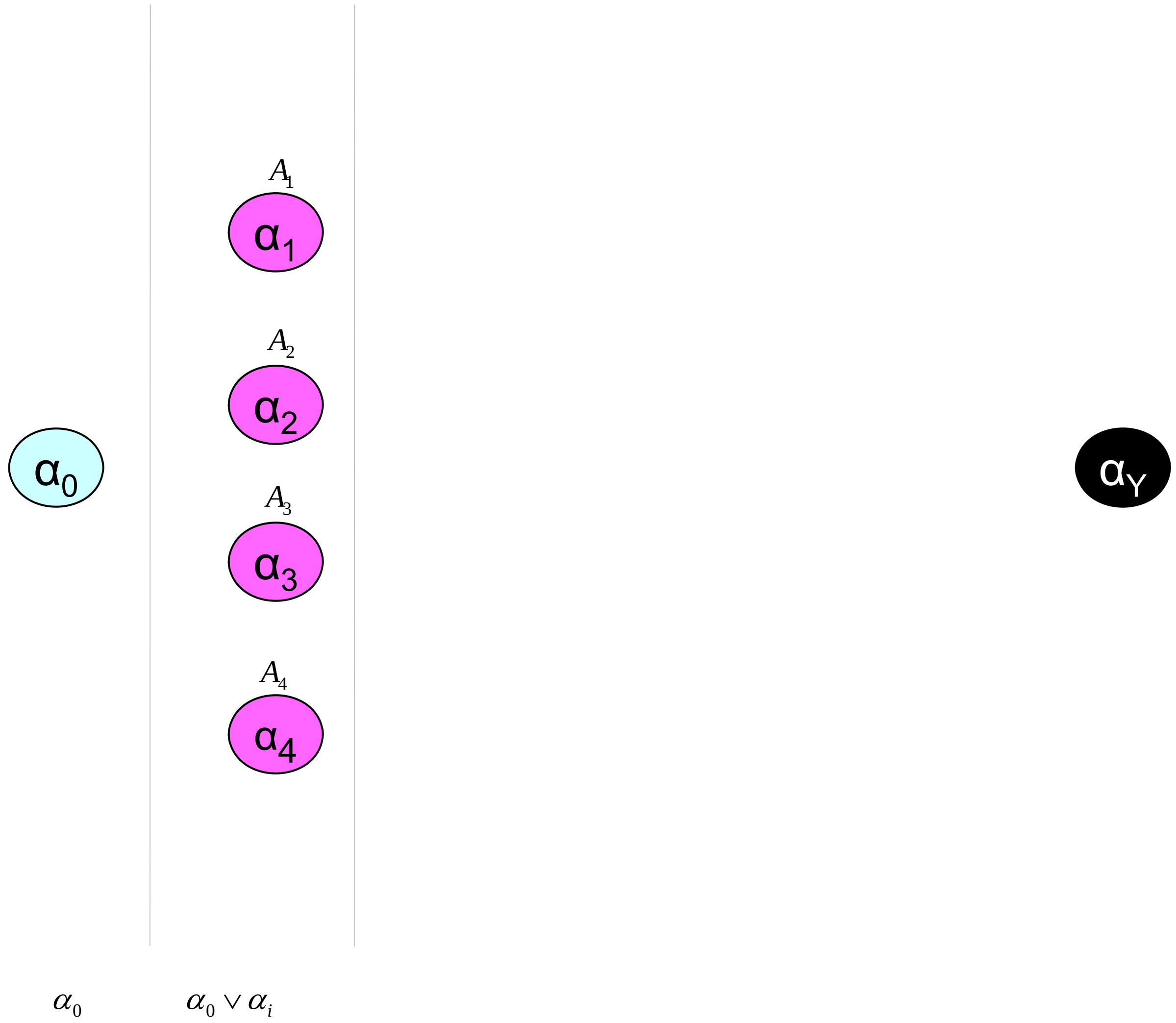
$w_1 = 0$; $w_2 = +1$;

distance2 = Conditional Entropy

The DID approach

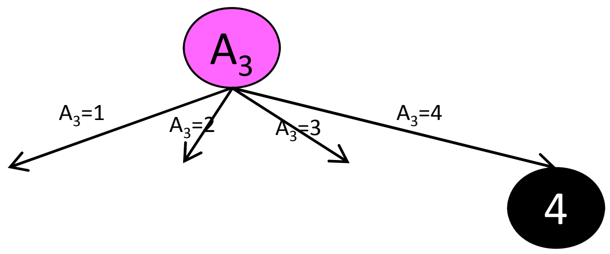


The DID approach

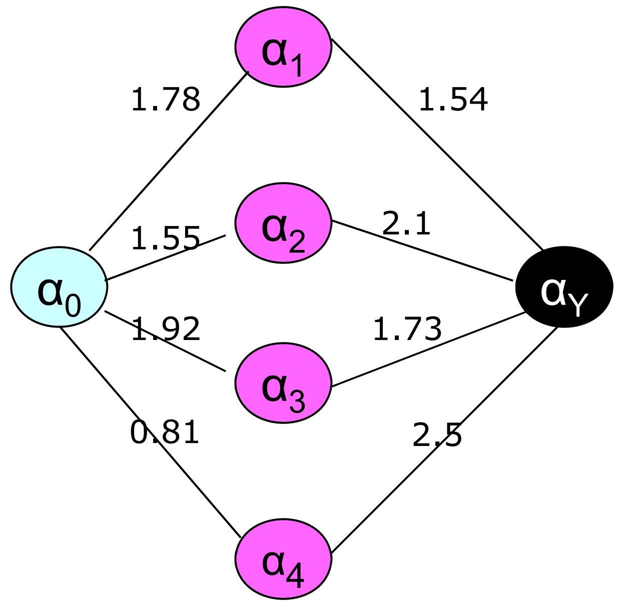


The DID approach

Classification Tree



Partitions Graph



Results for:

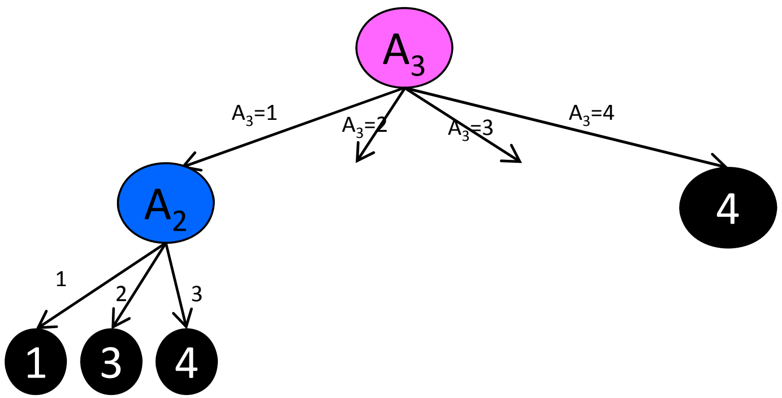
$w_1 = -2; w_2 = +1;$

$distance1 = Rokhlin(\alpha_i, \alpha_0)$

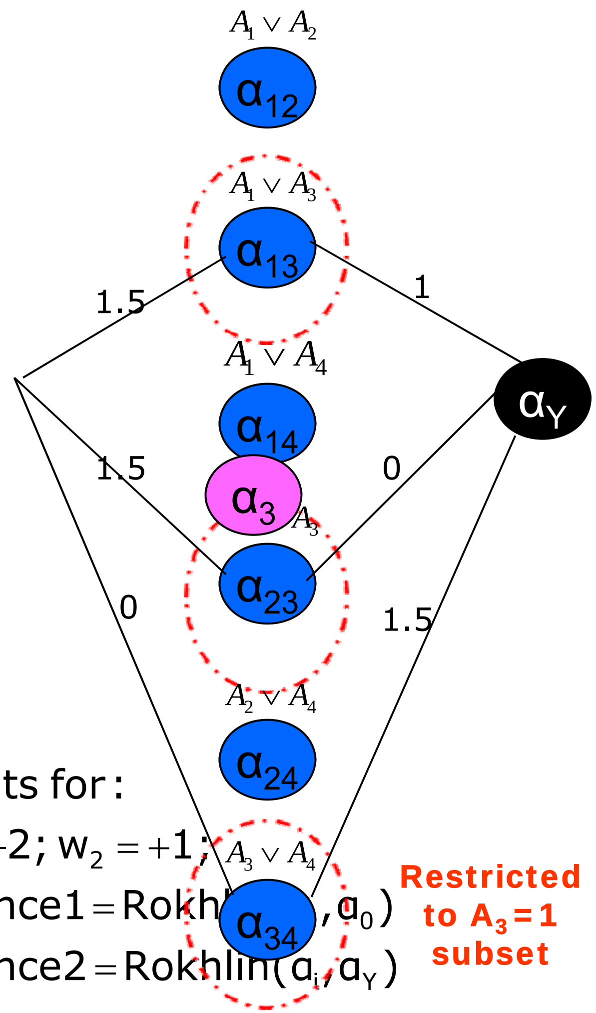
$distance2 = Rokhlin(\alpha_i, \alpha_Y)$

The DID approach

Classification Tree



Partitions Graph



Results for:

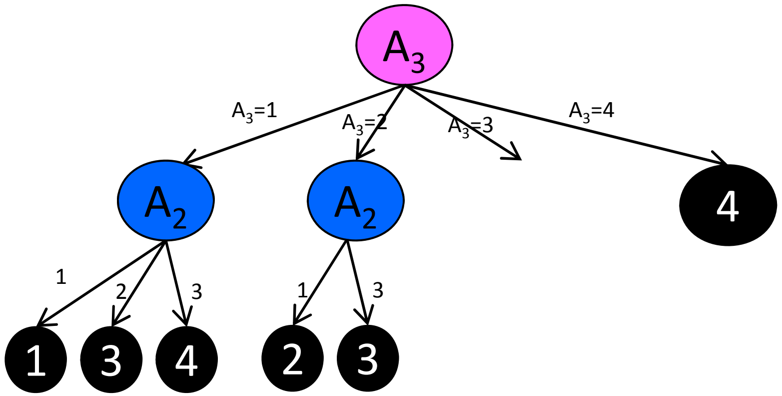
$w_1 = -2; w_2 = +1;$

$distance1 = Rokh$ $, \alpha_0)$

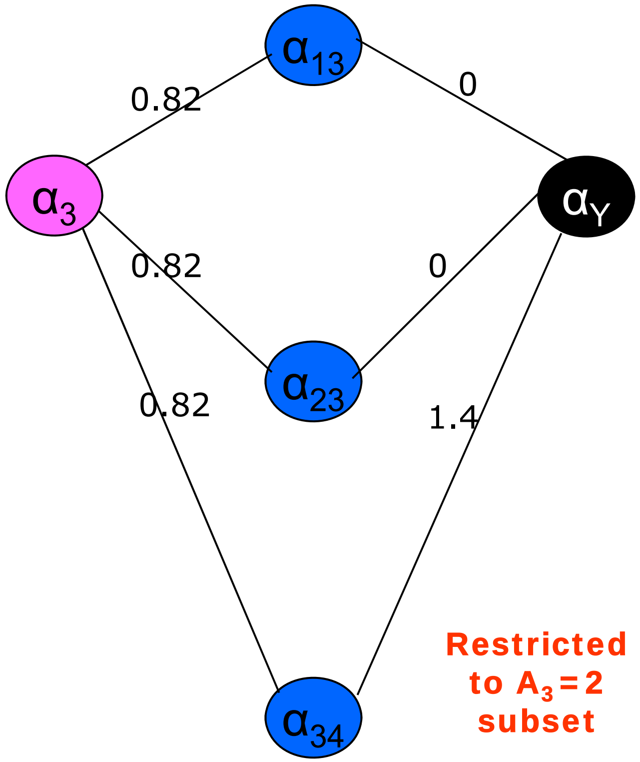
$distance2 = Rokhlin(\alpha_i, \alpha_Y)$

The DID approach

Classification Tree



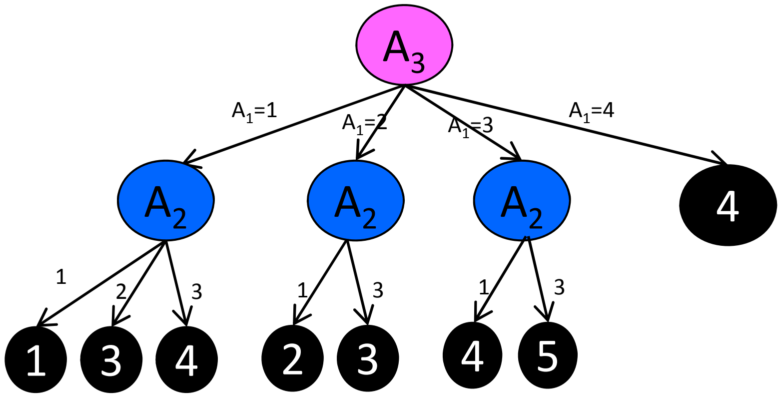
Partitions Graph



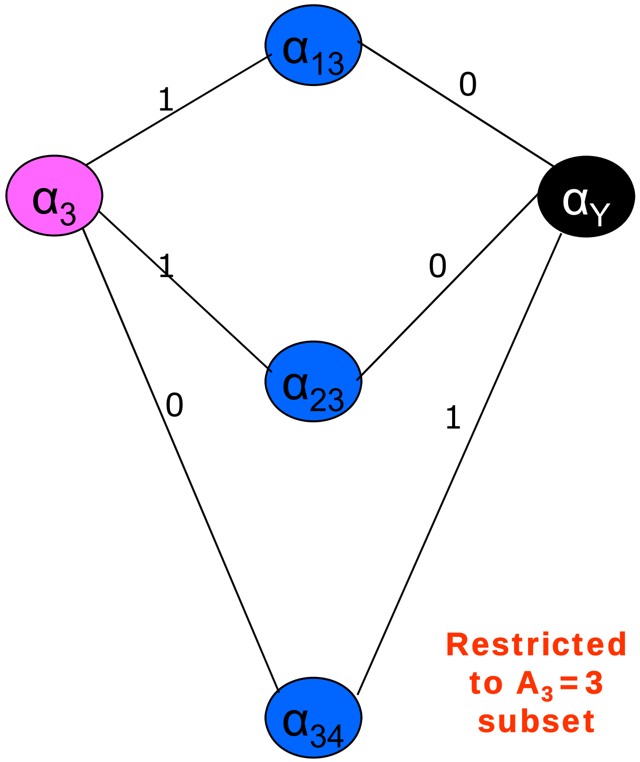
Restricted to $A_3=2$ subset

The DID approach

Classification Tree

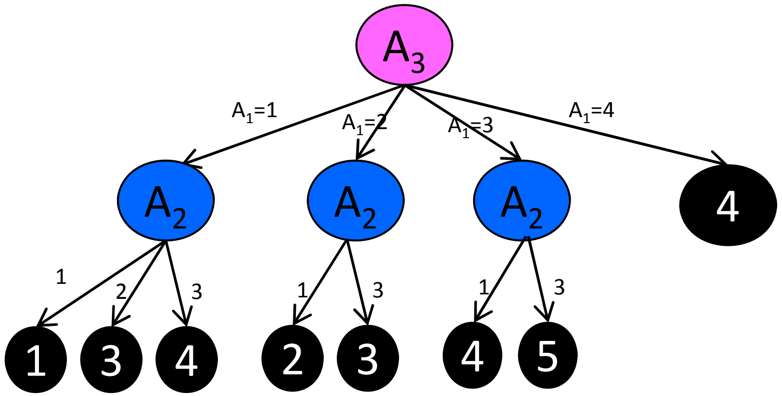


Partitions Graph

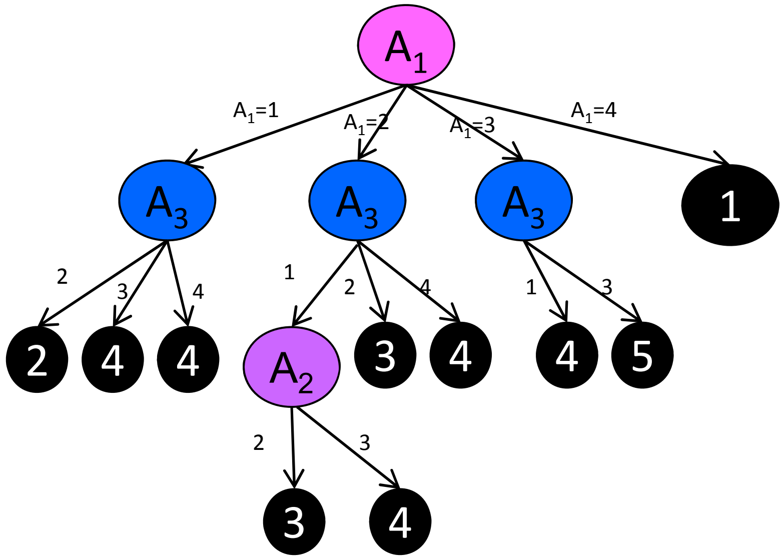


Comparing the Classification Trees

DD Tree



ID3/c4.5 Tree



Average depth = 1.8

No. of decision = 4

No. of leaves = 8

Max steps = 2

Average depth = 2.1

No. of decision = 5

No. of leaves = 10

Max steps = 3

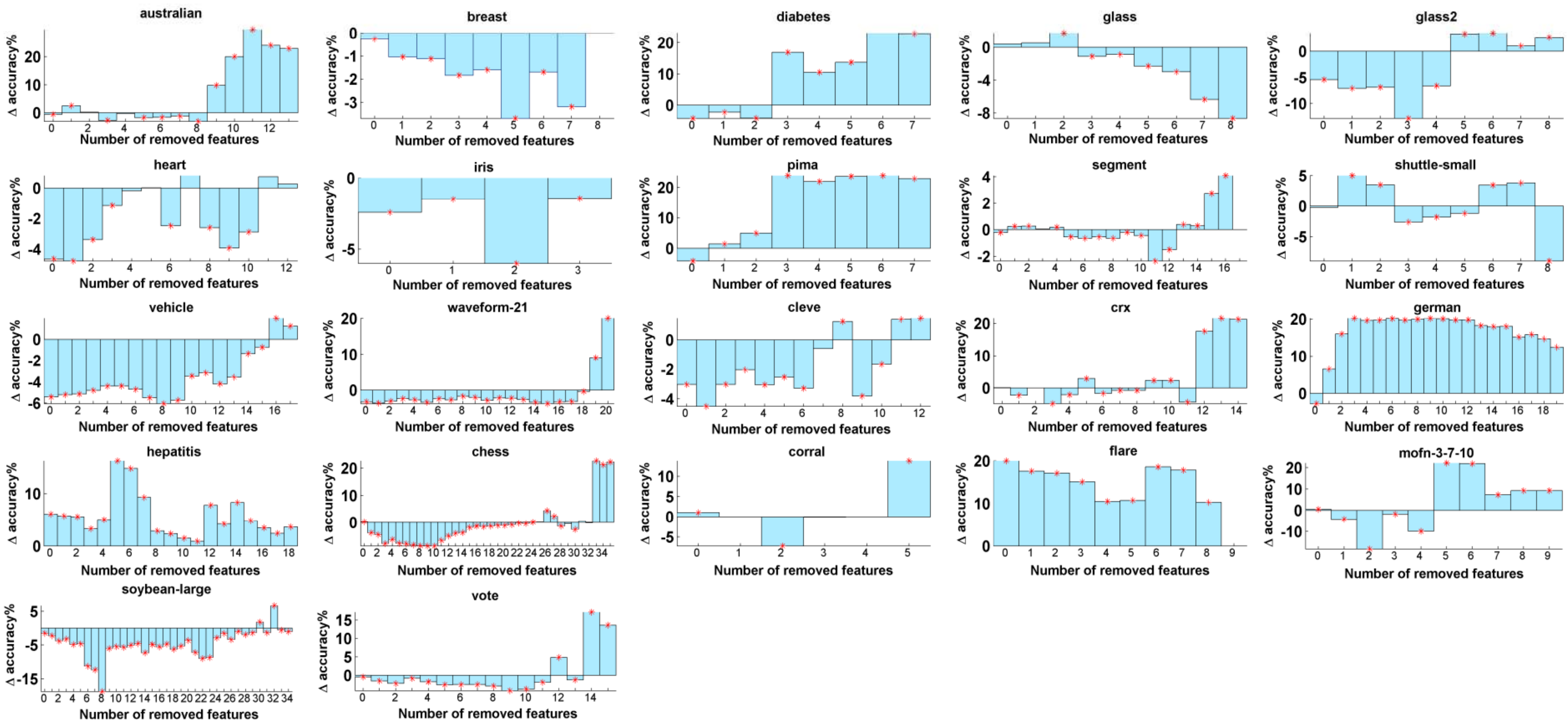
Some Results

Summarizing Comparison between ID3, C4.5 and DID decision trees

Dataset	Size		ID3		C4.5		DID	
	#instances	#Attributes	Average Depth	Accuracy	Average Depth	Accuracy	Average Depth	Accuracy
Monk's-1	124	6	3.21	82%	3.32	82%	2.66	96.%
Monk's-2	169	6	4.34	70.4%	4.6	75%	4.2	66%
Monk's full Random set	216	6	1.93	100%	2.04	100%	1.8	100%
Connect4	67,557	42	5.85	73.8%	10.16	79.4%	5.64	75%
SPECT Heart	80	22	9.6	75.1%	10.2	80.3%	9.3	76%
Voting	435	16	1.8	96%	2.2	96.6%	2.1	96%
Balance Scale	625	4	3.4	76.3%	3.4	78.6%	3.3	76.6%
Cars	1728	6	2.82	77.1%	2.83	77%	2.77	78.5%
Tic-Tac-Toe	958	9	4.62	80.6%	4.62	80.4%	4.6	76.2%
Soy Beans	47	35	1.35	100%	2.37	97%	1.32	97%
Lymphography	148	18	2.71	75.1%	6.51	77.3%	2.6	72.6%

Case	#features	SVM accuracy%	J48 %	DID accuracy %
australian	14	55.5	86.2	86.9
breast	9	96.5	93.6	93.5
diabetes	8	65.1	74.2	72.6
glass	8	69.16	50.1	51.8
glass2	8	76.68	75.3	82.1
heart	13	55.93	79.4	79.5
iris	4	96.67	94.4	95.6
pima	36	65.1	73.1	72.2
segment	18	63.9	94.1	93.6
Shuttle-small	9	89.41	62	61.9
vehicle	18	30.5	69.7	65.2
waveform-21	21	86.1	76.3	73.5
cleve	13	54.73	78.9	78
crx	15	65.67	87.5	87.6
german	20	70	65.2	66.6
hepatitis	19	83.55	57.4	64.2
chess	36	93.83	99.3	99.8
corral	6	96.89	98.1	98.3
flare	9	82.37	61.2	68.9
mofn-3-7-10	10	100	100	100
soybean-large	35	87.19	95.8	94.2
vote	15	95.35	94.7	95.4

Accuracy of C4.5 (J48) and DID as a function of the number of removed features for different cases taken from the UCI Repository



Outline

1. Introduction & Motivation
2. Our Partitions Approach
3. Example
4. Results
5. Mid-level solutions
6. Summary & Contribution

Summary & Contribution

Modeling the tree construction problem as a shortest path problem over a graph of partitions as nodes.

- A unified framework for existing DT algorithms
- Further Generalization via different metrics, e.g, Rokhlin, Entropy, etc. supported by IT
- Orthogonally vs. Information Gain
- Big Data fit: Shorter trees with smaller decisions for online scoring and recommendation

Thank you !
Questions?