集合知プログラミング
第7章　決定木によるモデリング(2)

06T4073R 三上健太

決定木の決定

ツリーをテキストで出力:ツリーを見る簡単な方法

しかし・・・
　ツリーが大きくなると目でたどるのが困難に・・・

↓

グラフィック表示(JPEG画像で表現)
　・テキスト表示は小さな決定木向け
　・グラフィック表示だと視覚的に見やすい

決定木の表示例(テキスト)

```
0:google?
T -> 3:21?
    T -> {'Premium': 3}
    F -> 2:yes?
            T -> {'Basic': 1}
            F -> {'None': 1}
F -> 0:slashdot?
    T -> {'None': 3}
    F -> 2:yes?
            T -> {'Basic': 4}
            F -> 3:21?
                    T -> {'Basic': 1}
                    F -> {'None': 3}
```

決定木の表示～グラフィック表現～

手順:

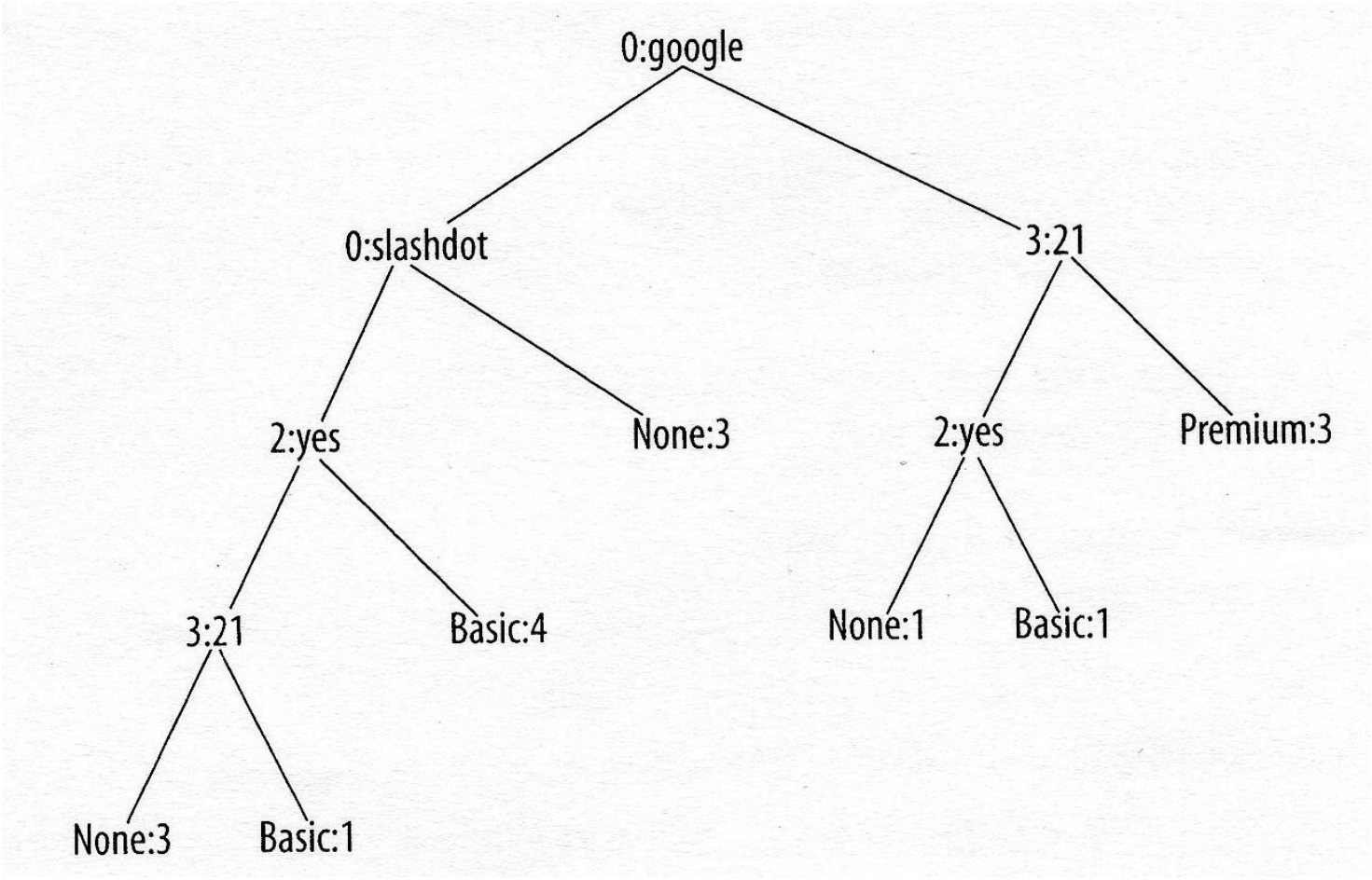
(1) 木を描画するのにどれだけスペースをとるか調べる
(キャンバスを作成)

(2) 各ノードに必要なスペースを計算、ノードごとに文字列や枝を追加していく

(3) (2)を葉にいきつくまで繰り返す

(4) 最終的に出来上がったものを画像として保存

作成された決定木例～グラフィック表示～



新しい観測を分類する

- 決定木に従って分類することが必要

- 手順:

(1) ルートノードからノードの条件をみる
(2) 条件に当てはまるほうの枝に進む
(3) 次のノードの条件をみる
(4) 上記手順を繰り返す
(5) 最終的に葉にたどり着けば分類完了

ツリーの刈り込み

- 過応答(overfit)問題
  - トレーニングデータに対して過剰に適合した状態

- 解決策

刈り込み(pruning)
  - ノードをまとめる作業。
  - 閾値を用いて合成できるノードは合成する

刈り込み結果例

```
0:google?
T -> 3:21?
    T -> {'Premium': 3}
    F -> 2:yes?
            T -> {'Basic': 1}
            F -> {'None': 1}
F -> 0:slashdot?
    T -> {'None': 3}
    F -> 2:yes?
            T -> {'Basic': 4}
            F -> 3:21?
                    T -> {'Basic': 1}
                    F -> {'None': 3}
```

```
0:google?
T -> 3:21?
    T -> {'Premium': 3}
    F -> 2:yes?
            T -> {'Basic': 1}
            F -> {'None': 1}
F -> {'None': 6, 'Basic': 5}
```

欠落データへの対処

- データセットに欠落がある場合がある
  ex. 所在地が識別できなかった etc...
  →このような場合も処理できるように決定木を改良

- どの枝をたどるべきかの判断時に欠落データが必要
  →両方の枝をたどることが可能
  →枝が持つ帰結を対等に扱わず、重み付けをする

欠落データへの対処(2)

- 決定木は全てが暗黙で1の重み付けを持っている
  (ある観測があるカテゴリに当てはまる見込みが1)
- 複数の枝を同時にたどるとき
  →各枝が持つ行数の割合に応じた重み付けをする