DRZEWA DECYZYJNE

Drzewa klasyfikacyjne – algorytm podstawowy

```
buduj_drzewo(S – przykłady treningowe, A – zbiór atrybutów) {
  utwórz węzeł t (korzeń przy pierwszym wywołaniu);
  if (wszystkie przykłady w S należą do tej samej klasy K)
      zamień t na liść z etykietą K;
  else {
    wybierz atrybut a ze zbioru A, który najlepiej klasyfikuje przykłady;
    przypisz węzłowi t test zbudowany na podstawie wybranego atrybutu a;
    for each wartość v_i atrybutu a {
      dodaj do węzła t gałąź odpowiadającą warunkowi a = v_i
      S_i = podzbiór przykładów z S, dla których a = v_i
      if S_i jest pusty
        dodaj do gałęzi liść z etykietą klasy, do której należy większość
        przypadków w S
      else
        buduj_drzewo(S_i, A-{a})
    }
  }
}
```

Algorytm ID3

- dla atrybutów
  - nominalnych
  - zdyskretyzowanych
- podział na podstawie Information Gain
  - faworyzacja atrybutów o dziedzinach wielowartościowych
  - wada: płaskie, szerokie drzewa np. przy wielu unikalnych wartościach
- lokalny wybór najlepszego atrybutu
- brak nawrotów
- dąży do jak najmniejszy drzew decyzyjnych (Ockham)

Entropia

- Entropia

$$Ent(S) = -\sum_{i=1}^{k} p_i \log_2 p_i$$

  - p_i to prawdopodobieństwo przynależności do klasy i-tej (estymowana przez n_i/n, gdzie n_i to liczba przykładów z klasą K_i, a n to liczba klas)
  - k to liczba klas
  - S to zbiór przykładów

- Klasyfikacja binarna

Wartość entropii	Opis
0	Przykłady tylko z jednej klasy
1	Po 50% przykładów z każdej klasy

- interpretacja: im mniejsza entropia tym więcej przykładów należy do jednej z klas

Information Gain

- Entropia warunkowa dla atrybutu *a*

$$Ent(S|a) = \sum_{j=1}^{p} \frac{n_{S_j}}{n} Ent(S_j)$$

  - *p* to liczba wartości atrybutu *a*
  - S_j to zbiór przykładów z wartością atrybutu v_j
  - ns_j – liczebność zbioru S_j
- Interpretacja: im mniejsza wartość entropii warunkowej tym większa jednorodność podziału
- Information Gain – ocena przyrostu informacji przy użyciu atrybutu *a*

$$Gain(S, a) = Ent(S) - Ent(S|a)$$

przyrost = entropia rodzica – suma ważonych entropii potomków

entropia rodzica powinna być duża, a suma ważonych entropii potomków mała (co oznacza dobrze odseparowane klasy)

Gain Ratio

- Problemy information gain:
  - preferuje atrybuty o dużej liczbie wartości
  - może prowadzić do przeuczenia
- Rozwiązanie: gain ratio
  - uwzględnienie rozmiarów i liczby potomków
  - kara dla atrybutów o dużych dziedzinach
- Używa split information do „normalizacji” przyrostu informacji

Gain Ratio - Split information

- split information (współczynnik podziału)

$$Split(S|a) = \sum_{j=1}^{p} \frac{|S_j|}{|S|} \log_2 \frac{|S_j|}{|S|}$$

  gdzie *p* to liczba partycji (liczba wartości *a*), natomiast *a* to wybrany atrybut dla podziału
- Duży split information = podobny rozmiar partycji
- Mały split information = niewielka liczba partycji zawiera większość przykładów
- Gain ratio:

$$GainRatio(S, a) = \frac{Gain(S, a)}{Split(S, a)}$$

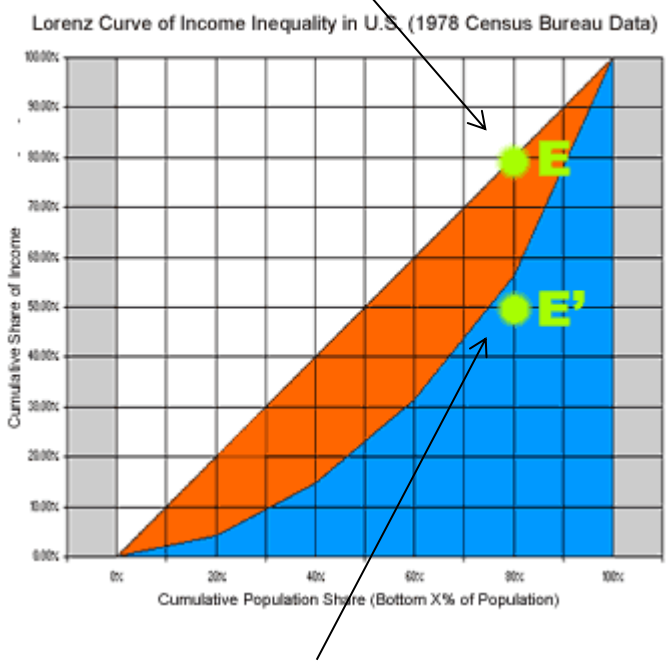
- wybieramy atrybut z największym gain ratio

może prowadzić do niezbalansowanych drzew

Gini index (1)

- Corrado Gini
- Interpretacja: stosunek obszaru pomiędzy krzywą Lorenza a prostą idealnego rozkładu do powierzchni całego obszaru pod prosta idealnego rozkładu
- może mierzyć nierównomierność rozkładu wartości atrybutu decyzyjnego wewnątrz węzła drzewa

80% ludzi posiada 80% dochodów



80% ludzi posiada 50% dochodów

Gini index (2)

- Formalnie współczynnik Giniego to:

  miara nierównomierności rozkładu zmiennej losowej

  $$Gini(S) = 1 - \sum_{i=1}^{m} {p_i}^2$$

  gdzie m to liczba klas, a p_i to prawdopodobieństwo, że przykład należy do klasy C_i
- Rozważany jest binarny podział na zbiory S_1 i S_2

  $$Gini(S,a) = \frac{|S_1|}{|S|} Gini(S_1) + \frac{|S_2|}{|S|} Gini(S_2)$$

  (ważona suma nieuporządkowania partycji)

  Ostatecznie

  $$\Delta Gini(S,a) = Gini(S) - Gini(S,a)$$
- Cechy
  - wartości z przedziału <0;1>, gdzie 0 oznacza pełną równomierność
  - nieprzystosowany dla dużej liczby klas
  - faworyzuje partycje o podobnych rozmiarach

C4.5

- Ulepszony ID3:
  - dopuszcza wartości numeryczne
  - dobrze radzi sobie z wartościami nieznanymi
  - wprowadza pruning dla radzenia sobie z szumem
- rozwojowa wersja to C4.8 (w wece J48)
- komercyjny następca to C5.0 (Rulequest)

C4.5 – testy dla węzła

- Trzy typy testów
  - dla danych dyskretnych po jednej krawędzi dla każdej wartości
  - dla danych ciągłych test binarny $a_i <= z$ i $a_i > z$, gdzie z to wartość progowa
  - dla danych dyskretnych podział wartości na grupy (po jednej gałęzi dla grupy)

C4.5 - ustalanie wartości progowej *z*

- trywialne podejście:
  - posortować wartości atrybutu, posortowana lista to $\{v_1, v_2, \ldots, v_m\}$
  - każda wartość pomiędzy v_i i v_{i+1} może być użyta do podziału (np. środek, ale w C4.5 wybierana raczej wartość v_i)
  - Przeglądniecie wszystkich *m-1* możliwych podziałów i obliczenie dla każdego z nich gain/gain ratio
- Są ulepszenia (np. Fayyad & Irani -> obliczanie entropii tylko dla par ze zmieniającymi się klasami)

C4.5 - pruning

- drzewo za bardzo dopasowuje się do danych uczących
  - overfitting
  - zbyt złożone drzewo
- Pruning – zastąpienie poddrzewa liściem, gdy stwierdzimy, że oczekiwany błąd poddrzewa jest większy niż liścia. Uwaga: nie testuj na zbiorze treningnowym – użyj hold out
- Rodzaje pruningu
  - prepruning – zatrzymaj podział liścia
  - postpruning – gdy całe drzewo gotowe usuń niepotrzebne części i zastąp je liśćmi

Prepruning

- Oparte na wynikach testów statystycznych
- Zatrzymaj podział, gdy nie ma statystycznie istotnego związku pomiędzy jakimkolwiek atrybutem a klasą w danym węźle
- test chi-kwadrat
- (uwaga: w ID3 używany test chi-kwadrat wraz information gain – tylko statystycznie istotne atrybuty były dostępne do wyboru przy podziale)
- Wada: czasami warunek stopu jest za ostry, ale zdarza się to rzadko
- Zaleta: jest relatywnie szybki

Postpruning

- Zbuduj całe drzewo
- Wykonaj pruning
  - subtree replacement – usuń węzeł i umieść tam liść

    Strategia bottom-up, sprawdź możliwość zastąpienia poddrzewa tylko, gdy drzewa poniżej już sprawdzone
  - subtree raising

    usuń węzeł (w środku) i rozdziel instancje do poniższych poddrzew (wolniejsze niż subtree replacement)

CHAID (1) – ogólna idea

- Chi-squared Automatic Interaction Detection
- do problemów klasyfikacji i regresji (tak jak CART)
- dla wersji klasyfikacyjnej podstawą podziału węzła jest test chi-kwadrat (a właściwie p-value z korektą Bonferroni)
- Podstawowa zaleta: automatyczny wybór n-kierunkowych podziałów węźle -> drzewa niebinarne
- wymaga dość dużych danych, gdyż n-kierunkowe podziały skutecznie redukują liczbę przykładów w liściach
- Liczba podziałów w węźle jest wyrażona liczbą Bella (suma liczby podziałów na dwie grupy plus suma podziałów na trzy grupy etc. aż do jednoelementowych grup)
- Dla 8 kategorii (unikalnych wartości w węźle) jest to 4139 podziałów – sprawdzanie wszystkich zbyt czasochłonne
- Kroki algorytmu to:
  - przygotowanie wartości predyktorów: predyktory, które nie są kategoryczne podlegają automatycznej dyskretyzacji częstościowej
  - **łączenie kategorii (i ich ponowne dzielenie)**
  - wybór predyktora do dzielenia

CHAID (2) – łączenie kategorii

- W iteracjach:
  - dla każdego predyktora wybierz parę wartości, które najmniej się różnią ze względu na zmienną zależną
  - wykonaj test chi-kwadrat dla takiej pary
  - jeżeli test daje wynik wskazujący na brak różnicy istotnej statystycznie dla łączenia (przy poziomie p) połącz te kategorie
  - powyższa czynność jest kontynuowana (aż do braku możliwości połączenia, poprzednio połączone kategorie mogą też być łączone)
  - grupy trzech lub więcej kategorii są sprawdzane, czy ich podział nie byłby lepszy (more significant)

RM: Decision Stump

- Tworzy drzewo decyzyjne na podstawie pojedynczego podziału
- n-krotne rozgałęzienia
- Najczęściej używany z AdaBoost
- Metody podziału
  - information gain
  - gain_ratio
  - gini_index
  - accuracy

RM: ID3

- implementacja zbliżona do oryginalnej propozycji Quinlana
- Parametry
  - criterion
  - minimal size for split
  - minimal leaf size
  - minimal gain
- Zalety
  - łatwa czytelność modelu
  - szybkość i nieduża wysokość modelu
- Wady
  - tendencja do przeuczenia dla małych zbiorów treningowych
  - tylko jeden atrybut jest testowany w danej chwili pod względem możliwości użycia do podziału

RM: Decision Tree

- działanie podobne do C4.5
- Parametry
  - *criterion* (information_gain, gain_ratio, gini_index, accuracy)
  - *apply pruning*
    - confidence – poziom ufności dla pessimistic error calculation for pruning
  - *apply prepruning*
    - *minimal gain* – minimalny zysk wymagany do przeprowadzenia podziału węzła
    - *minimal leaf size* – minimalna liczba przykładów w liściu
    - *minimal size for split* – minimalna liczba przykładów dla węzła by nastąpił podział
    - *number of prepruning alternatives* – gdy podczas prepruningu jakiś węzeł zostanie zablokowany przed podziałem, ile innych węzłów próbować podzielić
  - *maximal depth* - maksymalna wysokość drzewa (dla 1 generowany jest jeden podział)

RM: Decision Tree (weight-based)

- Operator złożony
- Umożliwia zdefiniowanie w jego wnętrzu metody wyznaczania atrybutów dla kolejnych węzłów drzewa

RM:CHAID

- W RapidMiner działa jak Decision Tree, a jedyną różnicą jest użycie testu chi-kwadrat zamiast information gain
- nie działa dla danych numerycznych
- Parametry
  - minimal size for split
  - minimal leaf size
  - minimal gain
  - confidence
  - maximal depth
  - number of prepruning alternatives

RM: MetaCost

- Operator złożony
- Pozwala na zdefiniowanie macierzy kosztów używanej przez algorytm będący wewnątrz operatora
- Parametry:
  - macierz kosztów
  - use subset for training
  - sampling with replacement

RM: Random Tree

- Działa podobnie jak C4.5
- Jedyna różnica to:

  Przy każdym podziale rozważany jest tylko losowo wybrany podzbiór atrybutów
- obsługuje dane nominalne i liczobowe
- Parametry takie jak dla Decision Tree oraz:
  - guess subset ratio – wybiera $\log(m) + 1$ atrybutów
  - subset ratio – ręczne ustawienie względnej liczby atrybutów

RM: Random Forest

- generuje zbiór drzew losowych, tj. z losowym doborem atrybutów dla podziału
- Decyzja podejmowana jest przez głosowanie
- Parametry takie, jak dla random tree oraz:
  - number of trees – liczba drzew do wygenerowania