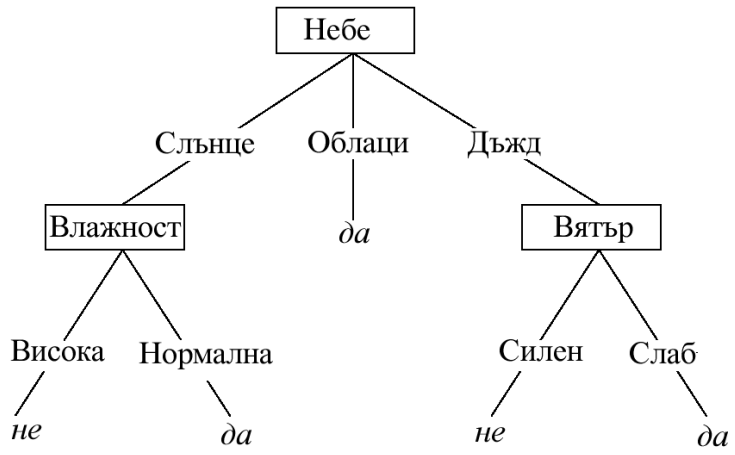
Лекция 3. Обучение чрез класификационни дървета

Научаването на понятия във вид на класификационни дървета е един метод за апроксимиране на целевата функция, приемаща дискретни стойности, в който тя се представя чрез класификационно дърво. За да станат по-разбираеми за човека, научените дървета могат да бъдат представени и по друг начин – като множества правила от вида АКО – ТО. Тези методи за самообучение са сред най-популярни индуктивни алгоритми и са били и продължават с голям успех да се използват за решаване на голям кръг практически задачи – от медицинската диагностика – до преценка за даване на заеми.

3.1. Представяне на класификационни дървета

Класификационните дървета класифицират примери, описани на езика Атрибут = Стойност. Класификационното дърво представлява една информационна структура, състояща се от възли, съединени със дъги – клонове. Всеки възел определя някой *тест* – проверка на стойността на определен атрибут от примера, а всеки клон, излизащ от този възел, съответства на една от възможни стойности на проверявания атрибут. Листата на класификационното дърво представляват стойностите (дискретни) на целевия атрибут. Класифицирането на новия пример започва от най-горния възел на дървото (коренът) и се осъществява чрез проверка за стойността на атрибута, описан в този възел; след това примерът се "пуска" надолу по клона, който съответства на конкретната стойност на проверявания атрибут в дадения пример. Описаният процес се повтаря в текущия възел докато примерът не стигне до някое от листата на дървото.



Класификационното дърво на понятието *Игра-на-тенис*

Фиг. 3-1.

За илюстрация да разгледаме наученото класификационно дърво на понятието *Игра-на-тенис*. Дървото се използва за предсказване, дали даденият ден, описан чрез свои атрибути *Небе, Температура, Влажност* и *Вятър*, е подходящ за игра на тенис (Фиг. 3-1).

Така примерът

<*Небе=Слънце,Температура = Горещо,Влажност=Висока,Вятър=Силен*>

ще бъде препратен долу по най-левия клон на това дърво и по тази причина ще бъде класифициран като отрицателен (*Игра-на-тенис=не*).

В общия случай, класификационните дървета представляват дизюнкция от конюнкции на ограничения, наложени на атрибутните стойности на примери. Всеки път от корена на дървото към някое от неговите листа съответства на конюнкцията на тестове върху атрибути, а самото дърво – на дизюнкцията на тези конюнкции. Например, показаното по-горе класификационно дърво съответства на израза:

$$(Небе = Слънце \wedge Влажност = Нормална)$$
$$\vee (Небе = Облаци)$$
$$\vee (Небе = Дъжд \wedge Вятър = Слаб)$$

3.2. За кои задачи са подходящи класификационни дървета

Научаването на понятия чрез класификационни дървета е най-подходящо за следните случаи:

- *Примерите са представени чрез двойки атрибут-стойност*. Примерите се описват с помощта на фиксирано множество от атрибути (напр. *Температура*) и техните стойности (напр. *Горещо*). Най-лесната ситуация за научаване на класификационни дървета е когато всеки атрибут може да приема неголям брой номинални (качествени) възможни стойности (напр. *Горещо, Топло, Студено*). Обаче, ще разгледаме и разширение на базовия алгоритъм, позволяващ работата с атрибути с непрекъснати стойности (напр., когато *Температура* се представя чрез числовата стойност).
- *Целевата функция приема дискретни стойности*. Показаното на рисунка класификационно дърво назначава Булева (двоична) класификация (т.е. *да* или *не*) на всеки пример. Класификационните дървета могат лесно да бъдат разширени за научаване на функции с повече от две възможни стойности. Едно по-съществено разширение на класификационните дървета (така наречени *регресионни дървета*) позволява да научават и функции с непрекъснатия диапазон от реални стойности.
- *Може да е необходимо дизюнктивното описание на научаваното понятие*. Както вече беше отбелязано, класификационните дървета по много естествен начин представят дизюнктивните понятия.
- *Обучаващите данни могат да съдържат грешки*. Методи за научаване на класификационни дървета са устойчиви на наличие на грешки в данни –

както на грешки в класификация на обучаващите примери, така и на грешки в стойностите на атрибути на тези примери.

- *Обучаващите данни могат да съдържат неизвестни (липсващи) стойности на атрибути*. Методите за научаване на класификационни дървета могат да се използват и в случаи, когато стойностите на атрибути в някои обучаващи примери са неизвестни (напр. стойността на *Влажност* за някои дни е неизвестна).

Много от реални задачи отговарят на тези параметри. По тази причина класификационните дървета са били прилагани към задачи за класифициране на пациенти в съответствие с тяхната диагноза, на оборудването – по техните повреди, на кандидатите за заеми - по вероятността за връщане на заема и т.н. Подобни задачи, в които целта е да класифицираш примери в една от възможни категории от едно предварително известно дискретно множество, често се наричат *класификационни задачи*.

3.3. Базовия алгоритъм за научаване на класификационни дървета

Разработени са различни алгоритми за научаване на класификационни дървета, обаче повечето от тях са варианти на базовия алгоритъм, който построява дървото от горе-надолу (от корена – към листата) и реализира евристичното претърсване на пространство от възможни класификационни дървета. Най-типичните представители на този подход са алгоритъм ID3 (Quinlan 1986) и неговият наследник C4.5 (Quinlan 1993), които ще бъдат основно разгледани в този курс. Ще разгледаме базовият алгоритъм, който приблизително съответства на ID3 (виж Табл. 2-1). Той описва начина на построяване на класификационното дърво, предназначено за научаване на Булевите функции, т.е. решаващо задачата за научаване на понятия.

ID3(Примери, Цел_атрибут, Атрибути)

Примери са обучаващите примери, *Цел_атрибут* е атрибутът, чиято стойност трябва да бъде предсказана, а *Атрибути* са останалите атрибути на примери, които се тестват от наученото класификационно дърво. Алгоритмът връща класификационното дърво, което коректно класифицира зададените *Примери.*

- **Създай** най-горен възел - *Корен* на дървото
- **Ако** всички *Примери* са положителни, **Върни,** като наученото, дърво с един единствен възел – *Корен*, маркиран със знака “+”.
- **Ако** всички *Примери* са отрицателни, **Върни,** като наученото, дърво с един единствен възел – *Корен*, маркиран със знака “-”.
- **Ако** *Атрибути* е празното множество, **Върни,** като наученото, дърво с един единствен възел – *Корен*, маркиран със знака, който съвпада с най-често срещано сред *Примери* значение на *Цел_атрибут*.
- **Иначе**

Започни

- $A \leftarrow$ този атрибут от *Атрибути*, който най-добре класифицира *Примери*
- Класификационен атрибут на *Корен* $\leftarrow A$
- За всяка възможна стойност v_i на A направи:
  - **Добави** в дървото новия клон под *Корен,* съответстващ на теста $A = v_i$
  - Нека *Примери*$(A=v_i)$ са подмножество от *Примери*, които имат стойността v_i на атрибута A
  - **Ако** *Примери*(v_i) е празното множество
    - **То** добави под този нов клон листо, маркирано със знака, който съвпада с най-често срещано сред *Примери* значение на *Цел_атрибут*
    - **Иначе** добави под този нов клон под-дърво ***ID3(****Примери*$(A=v_i)$*, Цел_атрибут, Атрибути – {A}))*

Край

- **Върни** *Корен*

Основната идея на алгоритъма е "разделяй и владей" – цялото множество от примери се разделя на по-малки множества, които се обработват по-лесно. По тази причина подобните алгоритми се наричат "разделящи". Построяването на дървото се започва от горе на долу с въпроса "кой атрибут трябва да бъде тестван в корена на дървото". За да намери отговора, всеки атрибут се оценява на базата на определен статистически показател (тест), определящ до колко добре този атрибут *самостоятелно* може да класифицира наличните обучаващи примери. Най-добрият атрибут се избира и се ползва като тест в корневия възел на дървото. За всяка възможна стойност на този атрибут се създава наследник на корена и обучаващите примери се сортират за всеки съответен възел-наследник (т.е. долу по клона, съответстващ на конкретната стойност на този атрибут в примерите). Целият процес се повтаря за всеки възел-наследник, използвайки само обучаващите примери, асоциирани с този възел, за да бъде избран нов най-добрия атрибут, който ще се служи като тест в съответния възел. По този начин се осъществява евристичното търсене за приемливо класификационно дърво, като алгоритмът никога не се връща за преразглеждане на вече приети по-рано решения.

3.3.1. Кой атрибут е най-добрия класификатор?

Най-важният момент в работата на ID3 е изборът, кой атрибут трябва да се тества във дадения възел. Естествено, бих ме искали да изберем такъв атрибут, който е най-полезен за класификацията на примери. Каква е добра количествена оценка за "полезност" на един атрибут? Ще дефинираме статистическата мярка, наречена "*информационна печалба*", която измерва доколко добре даденият атрибут разделя обучаващите примери в съответствие с тяхната целева класификация. На всяка

стъпка от процеса на изграждането на дървото ID3 използва тази мярка за информационната печалба за избор сред атрибути-кандидати.

3.3.1.1. Ентропия - мярка за еднородността на примери

За точното дефиниране на информационната печалба ще ни е необходимо да дефинираме една мярка, която в теорията на информация се нарича *ентропия* и характеризира (не)еднородността на произволен набор от примери. При зададеното множество от примери *S,* съдържащо положителни и отрицателни примери на някое целево понятие, ентропията на *S* относително тази двоична класификация се определя като:

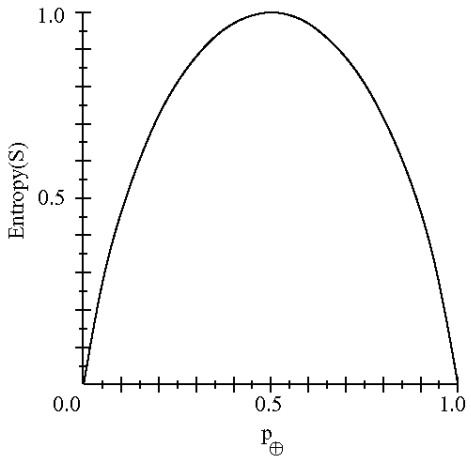
$$Entropy(S) \equiv -p_+ \log_2 p_+ - p_- \log_2 p_-$$

където p_+ е пропорцията на положителните примери в *S*, а p_- е пропорцията на отрицателните примери в *S*. Във всички изчисления ще определяме стойността на $0*\log_2 0$ като равна на 0.

За илюстрация да предположим, че *S* е колекцията от 14 примера на някое двоично понятие, която включва 9 положителни и 5 отрицателни примера (ще използваме означение *[9+,5-]* за описание на тази колекция). Ентропията на *S* относно тази двоична класификация ще бъде:

$$Entropy([9+,5-]) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.940.$$

Обърнете внимание, че ентропията е 0, ако всички членове на *S* принадлежат към един и същия клас. Например, ако всички членове са положителни $(p_+ = 1)$, то $p_- = 0$ и $Entropy(S) = -1*\log_2(1) - 0*\log_2(0) = 0.$ Ентропията е равна 1, когато броят на положителни и отрицателни примери в *S* е еднакъв. Ако колекцията съдържа неравен брой на отрицателни и положителни примери, то ентропията ще е между 0 и 1. Графиката на ентропията за двоичната класификация е показан на рисунката по-долу.



В теорията на информация ентропията се интерпретира като минималния брой на битове информация, необходими за предаване на съобщение, кодиращо класификацията на произволен член от S (т.е. член на S, избран по случаен начин с еднаква вероятност). Например, ако $p_+ = 1$, то получателят знае, че избраният пример ще е положителен, следователно никакво съобщение не трябва да се изпраща изобщо (и ентропията е равна на 0). Ако $p_+ = 0.5$, то за указание, че дадения пример е положителен или отрицателен е необходим 1 бит. Ако $p_+ = 0.8$, то една колекция от съобщения може да бъде кодирана, използвайки усреднено по-малко от 1 бит на съобщение, чрез назначаване на по-късите кодове на множеството от положителни примери и на по-дълги - на по-малко вероятни отрицателни примери.

В общия случай, когато целевият атрибут може да приема c различни стойности, ентропията на S относително c-ичната класификация се определя като:

$$Entropy(S) = -\sum_{i=1}^{c} p_i \log_2 p_i$$

където p_i е вероятността, че един случайно избран пример от S принадлежи към клас i. За оценка на тази вероятност може да се използва пропорцията на примери от този клас в S. Обърнете внимание, че логаритъмът остава с базата 2, тъй като ентропията е мярката на очакваната дължина на кодиране, която се мери в битове. Отбележете също така, че в този случай максималната стойност на ентропията е $log_2 c$.

3.3.1.2. Информационната печалба – мярка за очакваното намаляване на ентропията

След въвеждане на ентропията като мярка за еднородността на множество от обучаващите примери, сега можем да въведем и мярка за възможността на всеки атрибут да класифицира самостоятелно обучаващите примери. Тази мярка, наречена *информационна печалба*, е просто очакваното намаляване на ентропията, предизвикано от разделяне на примери в съответствие със стойностите на избрания атрибут. И така, информационната печалба *Gain(S, A)* на атрибута A относително множеството от примери S се определя като:

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Стойности(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

където *Стойности(A)* е множеството от възможни стойности на атрибута A, а S_v е подмножеството на S, в което всички примери имат стойността на атрибута A равна на v (т.е. $S_v = \{s \in S \mid A(s) = v\}$). Първият член на информационната печалба е ентропията на оригиналното множество S, а вторият – очакваната стойност на ентропията след като S ще бъде разделено чрез използване на атрибута A. Тази очаквана ентропия е просто сума от ентропиите на всяко подмножество S_v претеглена от пропорцията на примери $(|S_v|/|S|)$, принадлежащи към S_v.

Следователно *Gain(S, A)* е очакваното намаляване на ентропията, предизвикано от това, че на нас е известна стойността на атрибута *A*. Или, по друг начин, *Gain(S, A)* е информацията за стойността на целевата функция при известното значение на някой атрибут *A*. Стойността на *Gain(S, A)* е броят на битове, спестени при кодиране на целевото значение на произволен член от *S* след като стана известна стойността на атрибута *A*.

Да продължим с нашия пример за предпочитаните дни за игра на тенис. Един от атрибутите, описващи такива дни, е *Вятър*, който може да приема стойностите *Слаб* и *Силен* Ще продължим с множеството *S* от 14 примера *[9+,5-]*, обаче да смятаме, че 6 от положителните и 2 от отрицателните примера имат *Вятър = Слаб*, а останалите – *Вятър = Силен*. Информационната печалба от сортиране на началните 14 примера по атрибута *Вятър* може да се изчисли по следния начин:

$$Стойности(Вятър) = \{Слаб, Силен\}$$

$$S = [9+,5-]$$

$$S_{Слаб} = [6+,2-]$$

$$S_{Силен} = [3+,3-]$$

$$Gain(S, Вятър) = Entropy(S) - \sum_{v \in \{Слаб, Силен\}} \frac{|S_v|}{|S|} Entropy(S_v) =$$

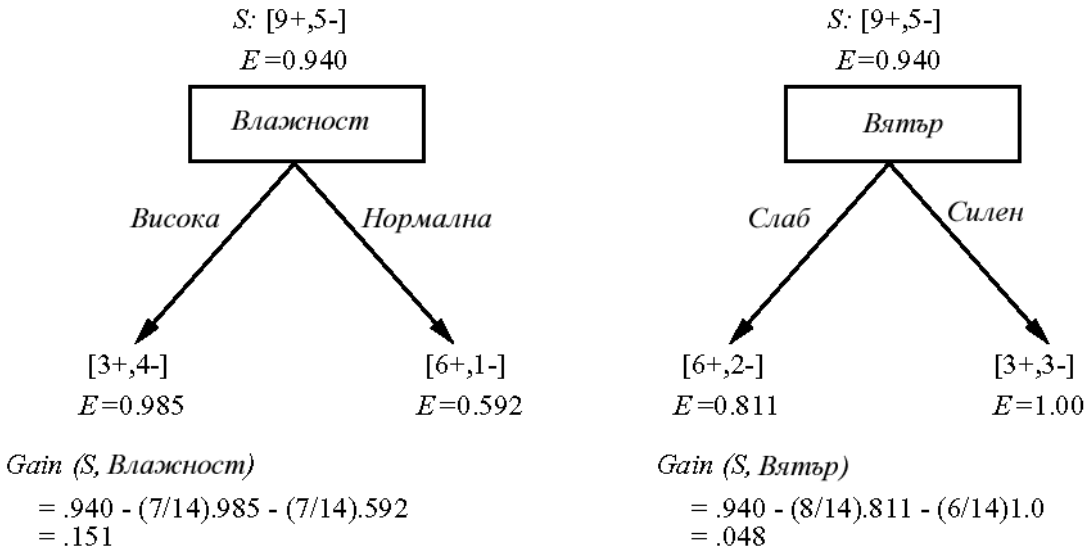
$$= Entropy(S) - (8/14) Entropy(S_{Слаб}) - (6/14) Entropy(S_{Силен}) =$$

$$= 0.940 - (8/14)0.811 - (6/14)1.0 = 0.048$$

ID3 използва информационната печалба за избор на най-добрия атрибут на всяка стъпка от построяването на дървото. Използването на тази информация е резюмирано на следната рисунка, сравняваща информационната печалба от използване на два атрибута *Вятър* и *Влажност*. Обучаващите примери са зададени в Таблица 3.2.

Ден	Небе	Температура	Влажност	Вятър	Игра-на-тенис
D1	Слънце	Горещо	Висока	Слаб	не
D2	Слънце	Горещо	Висока	Силен	не
D3	Облаци	Горещо	Висока	Слаб	да
D4	Дъжд	Топло	Висока	Слаб	да
D5	Дъжд	Студено	Нормална	Слаб	да
D6	Дъжд	Студено	Нормална	Силен	не
D7	Облаци	Студено	Нормална	Силен	да
D8	Слънце	Топло	Висока	Слаб	не
D9	Слънце	Студено	Нормална	Слаб	да
D10	Дъжд	Топло	Нормална	Слаб	да
D11	Слънце	Топло	Нормална	Силен	да
D12	Облаци	Топло	Висока	Силен	да
D13	Облаци	Горещо	Нормална	Слаб	да
D14	Дъжд	Топло	Висока	Силен	не

Таблица 3.2. Обучаващите примери на понятието *Игра-на-тенис*



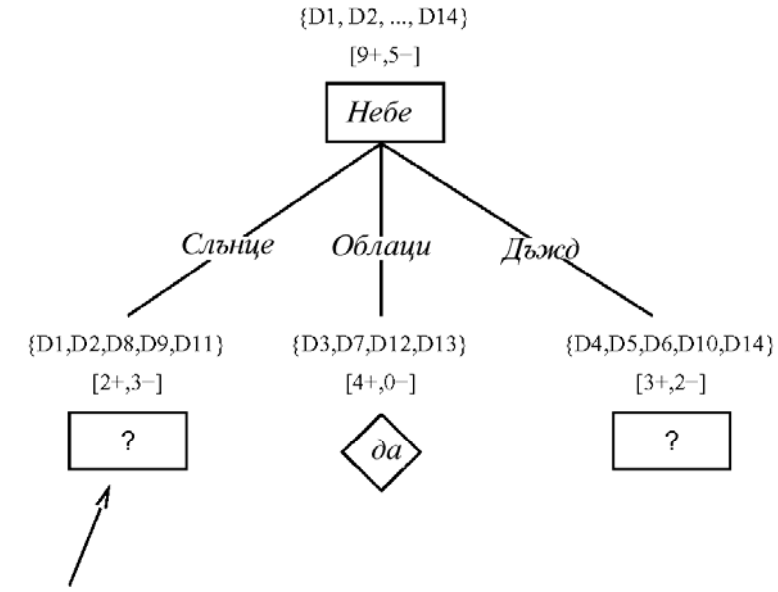
Както се вижда от стойностите на информационната печалба, по-добрият атрибут е *Влажност*.

3.3.2. Илюстративен пример

Да разгледаме по-подробно работата на ID3 на примера, представен с обучаващите примери от Таблицата 3.2. Целевият атрибут е *Игра-на-тенис*, чиито стойностите *да* или *не* трябва да бъдат предсказани по стойностите на атрибутите *Небе, Температура, Влажност и Вятър*. Да разгледаме първата стъпка на алгоритъма, при която се създава корневият възел. Кой атрибут трябва да се тества пръв? За да отговори на този въпрос ID3 изчислява информационната печалба за всеки от четирите атрибута, след което избира този с най-голямата печалба. След изчисленията получаваме (*S* – означава 14 примера от Таблицата 3.2):

Gain(S, Небе) = 0.246; *Gain(S, Влажност) = 0.151;*
Gain(S, Вятър) = 0.048; *Gain(S, Температура) = 0.029*

Виждаме, че съгласно мярката за информационната печалба, най-доброто предсказване на целевия атрибут върху обучаващите примери дава атрибутът *Небе*. По тази причина той се избира като тест за корневия възел, а под него се създават клонове, отговарящи на всички негови възможни стойности *(Слънце, Облаци, Дъжд)*. Полученото частично класификационно дърво е показано на следващата рисунка, в която са представени и обучаващи примери, сортирани по всеки от клонове. Обърнете внимание, че всеки обучаващ пример, за който *Небе = Облаци* е положителен пример на понятието *Игра-на-тенис*. По тази причина този клон се завършва с възел, който е листо на дървото с класификацията *Игра-на-тенис = да.* Останалите два възли-наследници, съответстващи на *Небе = Слънце* и *Небе = Дъжд,* продължават да имат ненулева ентропия, което означава, че класификационното дърво трябва да бъде продължено от тези възли надолу.



$S_{Слънце}$ = {D1,D2,D8,D9,D11}

$Gain\ (S_{Слънце}, Влажност) = .970 - (3/5)\ 0.0 - (2/5)\ 0.0 = .970$

$Gain\ (S_{Слънце}, Температура) = .970 - (2/5)\ 0.0 - (2/5)\ 1.0 - (1/5)\ 0.0 = .570$

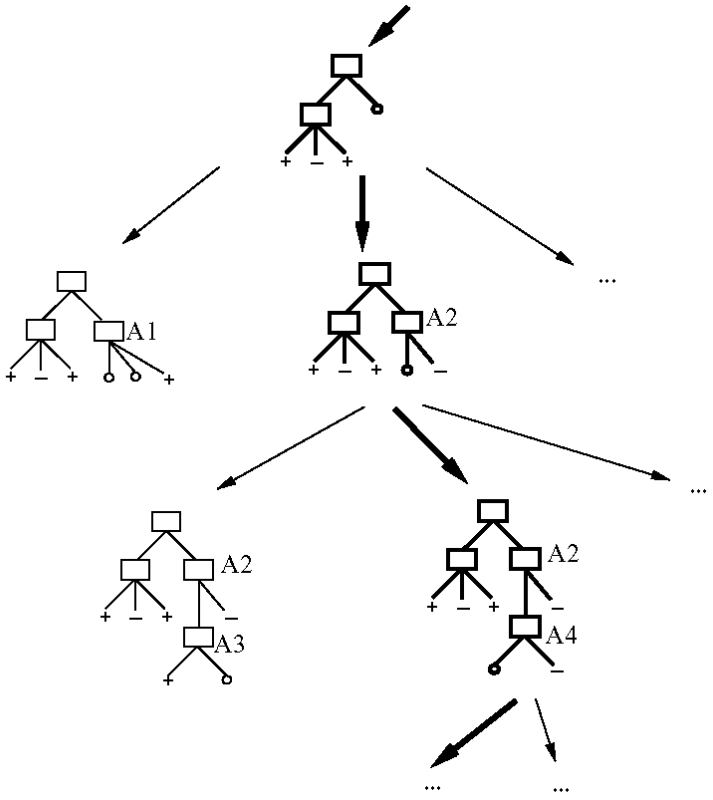
$Gain\ (S_{Слънце}, Вятър) = .970 - (2/5)\ 1.0 - (3/5)\ .918 = .019$

Процесът на избора на нов атрибут и разбиването на обучаващите примери на подгрупи се продължава във всеки нетерминален възел, като се използват само тези обучаващите примери, които са асоциирани с конкретния възел. Атрибутите, които вече били използвани по-горе в дървото, се премахват от разглеждането, така че всеки атрибут може да се среща *само веднъж във всеки път по дървото*. Процесът се продължава за всеки по-долен възел, докато не бъде изпълнено едно от следните два условия: 1) всеки от наличните атрибути вече е включен в някой възел от дървото по този път или 2) всички обучаващите примери, асоциирани с този възел-листо имат едно и също значение на целевия атрибут (т.е. тяхната ентропия е нула).

3.4. Обучение чрез класификационни дървета като търсене в пространството на хипотези

Както всеки един метод за самообучение чрез индукция ID3 може да бъде характеризиран като търсене в пространството на хипотези на такава хипотеза, която съвместима с обучаващите примери. Пространството на хипотези,

претърсвано от ID3, е множеството от възможни класификационни дървета. ID3 изпълнява евристичното (от *простото-към-сложното*) търсене в пространството на хипотези като започва с празното дърво, а след това постепенно усложнява хипотези в търсене на класификационното дърво, което коректно класифицира обучаващите данни. Оценъчната функция, която направлява това търсене, е мярката за информационната печалба (виж рисунката по-долу)



Разглеждайки ID3 в термините на неговото пространство на търсене и използваната стратегия за търсене, можем да добием представа за възможностите и ограничения на този алгоритъм:

- Пространството на хипотези на ID3 се състои от всички класификационни дървета и е *пълното* пространство на функции с дискретни стойности, относително наличните атрибути. Тъй като всяка функция с дискретни стойности може да бъде представена като класификационно дърво, ID3 избягва един от главните рискове на методите, претърсващи непълното пространство от хипотези (като, например, методи, разглеждащи само конюктивните хипотези): това, че пространството на хипотези може и да не съдържа целевата функция.
- При търсене в пространството от възможни класификационни дървета ID3 обработва само една единствена текуща хипотеза. Това контрастира, например, с по-рано разгледания метод за елиминиране на кандидати в пространството на версии, който обработваше множеството от *всички* хипотези, съвместими с налични обучаващи данни. Избирайки само една

единствена текуща хипотеза, ID3 губи възможности, произтичащи от явното представяне на всички съвместими хипотези. Например, алгоритмът не може да определи, колко алтернативни дървета, съвместими с наличните обучаващите данни, могат да бъдат построени или да дава заявки (за нови примери), които позволяват оптималния избор между състезаващи се хипотези.

- В своята чиста форма ID3 никога не се връща при търсенето. След като веднъж избере някой атрибут за тестване на определеното ниво от дървото, алгоритмът никога не се връща да преразгледа този свой избор. Това, естествено, води до обичайни рискове на градиентното търсене без възврат – сходимостта към *локално оптимални* решения, които не са оптимални глобално. В случая на ID3 локалното оптимално решение съответства на класификационното дърво, което е било избрано от алгоритъма при претърсване на един единствен път в общото пространство от възможни дървета. Обаче, това дърво може да се окаже по-малко ефективно от другите дървета, които би могли да бъдат срещнати при преглед на други възможни пътища на търсене. По-късно ще разгледаме едно разширение на базовия алгоритъм, който добавя възможността за възврат в една определена форма (допълнително подрязване на дървета).
- На всяка стъпка от търсенето ID3 използва *всички* обучаващи примери, за да получи статистически обосновани решения за това, как да направи по-съвършена текущата хипотеза. Това контрастира с методите за инкременталното взимане на решения на базата на отделни обучаващи примери (като това става при FIND-S и алгоритъма за елиминиране на кандидати). Едно от предимствата при използване на статистическите свойства на всички примери (напр. информационната печалба) е че полученото търсене е по-малко чувствително към грешки в отделни обучаващи примери. По тази причина ID3 може лесно да бъде разширен, за да работи със зашумени обучаващи данни чрез модификация на критерия за край, като да приема хипотезите, които непокриват абсолютно точно обучаващите данни.

3.5. Индуктивното пристрастие на обучение чрез класификационните дървета

Каква е политиката, която се ползва от ID3 при правене на обобщение от обучаващите примери с цел класификация на нови примери? С други думи, какво е неговото индуктивно пристрастие? Напомням, че индуктивното пристрастие е минималното множество от предположения, които заедно с обучаващите данни дедуктивно потвърждават класификации, назначавани от алгоритъма на нови, неизвестни примери.

При зададеното множество от обучаващи примери обикновено съществуват няколко различни класификационни дървета, съвместими с тези примери. Следователно, описанието на индуктивното пристрастие на ID3 се състои във

формулирането на тези предпочитания, които той използва, за да избира една от възможностите. ID3 избира първото приемливо дърво, което той среща в процеса на свое градиентно, от простото-към-сложното търсене в пространството от възможни дървета. Трудно е точно да се опише индуктивното пристрастие на този алгоритъм поради сложното взаимодействие между евристиката, използвана за избор на атрибути, и конкретни обучаващи примери, използвани от алгоритъма. Обаче, това пристрастие може да се формулира приблизително по следния начин:

Късите дървета се предпочитат пред дългите. Дърветата, които поместват атрибути с висока информационна печалба по-близо до корена си, се предпочитат пред тези, които не го правят.

3.5.1. Ограничително пристрастие и избирателно пристрастие

Има едно интересно различие между двата вида на индуктивното пристрастие, проявяваните от ID3 и алгоритъма за елиминиране на кандидати. Да разгледаме по-внимателно разликата между търсене в пространството от хипотези, изпълнявано от тези два алгоритъма:

- ID3 претърсва *пълното* пространство от хипотези (т.е. е способен да изрази произволна целева функция с дискретни стойности). Той използва метода за *непълно* (евристично) търсене, от по-прости хипотези към по-сложни, докато не бъде намерена хипотеза, отговаряща на нужните условия (съвместима с всички обучаващи данни). Индуктивното пристрастие на ID3 е прякото следствие от подреждането на хипотези, получавано от използваната стратегия за търсене. Пространството на хипотези не налага никакви допълнителни предпочитания.
- Алгоритмът за елиминиране на кандидати претърсва *непълното* пространство от хипотези (т.е. което съдържа само определено подмножество от всички възможни за научаване понятия). Той използва метода за *пълно* търсене, намирайки всички хипотези, съвместими с обучаващите данни. Индуктивното пристрастие на алгоритъма за елиминиране на кандидати е прякото следствие от изразителната сила на използвания начин за представяне на хипотези. Стратегията за търсене не налага никакви допълнителни предпочитания.

На кратко, индуктивното пристрастие на ID3 следва от използваната *стратегия за търсене*, докато индуктивното пристрастие на алгоритъма за елиминиране на кандидати следва от дефиницията на неговото *пространство на търсене.*

Индуктивното пристрастие на ID3 описва начина за предпочитане на едни хипотези пред други, без никакви тежки ограничения на самите хипотези. Тази форма на пристрастие обикновено се нарича *избирателно пристрастие (*или *пристрастие при търсене).* Пристрастието на алгоритъма за елиминиране на кандидати е една форма на ограничения върху множество от разглежданите хипотези. Тази форма на пристрастие се нарича *ограничително пристрастие (*или *езиково пристрастие)*.

Знаейки, че за да бъде направено някое обобщение извън обучаващите примери е необходимо наличието на индуктивното пристрастие, коя неговата форма – избирателна или ограничителна – е за предпочитане? Обикновено, за предпочитане е избирателното пристрастие, тъй като позволява на алгоритъма да работи в пълното пространство от хипотези, осигурявайки по този начин, че истинското описание на неизвестна целева функция се съдържа в това пространство.

3.5.2. Защо се предпочитат по-къси хипотези?

Дали индуктивното пристрастие на ID3, предпочитащо по-къси дървета, представлява един действително обоснован базис за обобщаване извън сферата на обучаващите примери? В по-общ контекст нашият въпрос се свежда до следното: "Дали по-простото нещо е за предпочитане от по-сложното?" Този въпрос се дебатира от философите вече от много столетия и още до сега е останал не решен. Уилиям Окам е бил един от първите, обърнал внимание на този въпрос (около 1320 г.), така че това пристрастие често се нарича "Бръснача на Окам": "*Nunquam ponenda est pluralitas sin necesitate*" – "Не умножавай същностите повече от необходимото", което в нашия случай може да са интерпретира като:

"За предпочитане е най-простата хипотеза, обясняваща данни".