Ekspertski Sistemi Vežbe

Stabla odlučivanja

Program vežbi

- Algoritmi pretraživanja
- Formalna logika (model predstavljanja znanja)
- Teorija igara
- Semantičke mreže i okviri
- Produkcioni sistemi
- Strategije rešavanja problema
- Stabla odlučivanja

Šta je učenje

- Herbert Simon: “Učenje je svaki proces kojim sistem unapređuje performanse na osnovu iskustva.”
- Šta je zadatak?
  - Klasifikacija
  - Rešavanje problema / planiranje / upravljanje

Zašto je potrebno mašinsko učenje (Machine Learning)?

- Rešavanje problema klasifikacije
- Učenje modela podataka
- Razumevanje i poboljšanje efikasnosti ljudskog učenja
- Otkrivanje novih odnosa i struktura koje nisu poznate ljudima (“data mining”)
- Popunjavanje nekompletne specifikacije o domenu

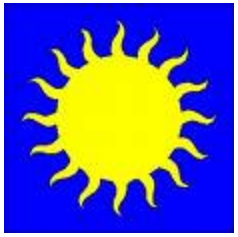
Klasifikacija

- Dodeliti objekat/događaj jednoj od datih kategorija iz konačnog skupa kategorija.
  - Medicinska dijagnostika
  - Upoterba / izdavanje kreditnih kartica
  - Detekcija prevara u elektronskom poslovanju
  - Otkrivanje virusa
  - Filtriranje spam emailova
  - Preporučeni sadržaj na Internetu
    - Knjige, filmovi, pesme, ...
  - Investicije
  - DNK
  - Prepoznavanje govora
  - Prepoznavanje rukopisa
  - Prepoznavanje slika

Stablo odlučivanja (Decision Tree)

- *Stablo odlučivanja je stablo u kojem:*
  - Svakom čvoru koji nije list je pridružen atribut
  - Svakom listu je pridružena klasifikacija, obeležje klase (class label, npr. + ili -, *da* ili *ne*)
  - Svakoj grani je dodeljena jedna vrednost atributa od kojeg grana počinje

PRIMER



Upotreba stabla odlučivanja

- Stablo odlučivanja se koristi kao klasifikator tako što uzima ulazni primer, koji je dat kao vektor atributa, i::
  1. Atribut koji je koreni čvor se interpretira kao pitanje, a odgovor se određuje na osnovu vrednosti tog atributa iz ulaznog primera
  2. Odgovor određuje koji čvor naslednik se posećuje
  3. Ponavljati sve dok se ne dođe do lista.
     Oznaka klase lista je klasifikacija ulaznog primera

Učenje stabla odlučivanja

- Koje je najbolje stablo odlučivanja?
- **Ockham's Razor**
  - ***Najjednostavnija hipoteza*** koja je konzistentna sa svim opservacijama je najčešće najbolji izbor
  - Najbolje je ***najmanje stablo odlučivanja*** koje ispravno klasifikuje sve trening primere
- Nalaženje dokazano najmanjeg stabla je NP-hard problem, pa je dovoljno konstruisati "prilično malo" stablo

Konstruisanje stabla odlučivanja upotrebom Greedy algoritma

- **ID3** ili **C5.0**
- Top-down konstrukcija stabla odlučivanja:
  1. Izabrati „najbolji atribut“ za novi čvor na trenutnom nivou u stablu
  2. Za svaku moguću vrednost izabranog atributa:
     a) Podeliti primere upotrebom mogućih vrednosti ovog atributa, i dodeliti te podskupove primera odgovarajućem čvoru nasledniku
     b) Rekurzivno generiši naslednika sve dok (u idealnom slučaju) svi primeri čvora ne budu svi + ili -

DECISION TREE ALGORITAM

buildtree(*examples*, *attributes*, *default*)
/* *examples*: lista trening primera
* *attributes*: skup kandidat pitanja,
* npr. „koja je vrednost atributa x_i?”
* *default*: podrazumevana vrednost klase predikcije,
* npr. rezultat većinskog glasanja*/

IF empty(*examples*) THEN return(*default*)
IF (*examples* have same label *y*) THEN return(*y*)
IF empty(*attributes*) THEN return(majority vote in *examples*)
q = best_attribute(*examples*, *attributes*)
Neka postoji ***n*** mogućih vrednosti atributa ***q***

- Kreiraj i vrati unutrašnji čvor sa **n** naslednika
- **n**-ti naslednik je kreiran pozivom:
  buildtree({*example* | **q=n**-ta vrednost}, *attributes*-{**q**}, *default*)

Decision Tree algoritam

- Kako izabrati „najbolji atribut“?
  - **Random:** slučajnim izborom izabrati bilo koji atribut
  - **Least-Values:** izabrati atribut koji ima najmanji broj mogućih vrednosti
  - **Most-Values:** izabrati atribut sa najvećim brojem mogućih vrednosti
  - **Max-Gain:** izabrati atribut koji ima najveću očekivanu dobit informacija (information gain)

Information Gain

- Kako se određuje information gain?
  - **cilj:** pokušati izabrati atribut koji će rezultovati najmanjim podstablima koja počinju u njegovim naslednicima
  - koristi teoriju informacija (information theory)

Information Theory

- Koliko da/ne pitanja očekujete da pitate radi određivanja broja koji sam zamislio u opsegu od 1 do 100?
- 7
- Sa svakim da/ne pitanjem u optimalnom stablu odlučivanja najviše 1/2 od preostalih elemenata može biti eliminisana

- $log_2 100 = 6.64$

Information Theory

- Ako je dat skup S veličine $|S|$, očekivan broj pokušaja radi određivanja konkretnog elementa je:
- $log_2 |S|$
- Neka ova vrednost bude vrednost informacije saznanja, a da nismo ni morali da postavljamo pitanja

Entropija

- Entropija skupa primera, S, za binarnu klasifikaciju (ishod ima samo dve klase) je:

$$Entropy(S) = -p_1 \log_2(p_1) - p_0 \log_2(p_0)$$

gde je p_1 procenat pozitivnih primera u S, a p_0 procenat negativnih.

- Ako su svi primeri isti (pripadaju istoj klasi) entropija je nula (korisnimo $0 \cdot \log(0)=0$)
- Ako su svi primeri podjednako raspodeđeni ($p_1=p_0=0.5$), entropija ima maksimalnu vrednost, 1.
- Entropija se može posmatrati kao prosečan broj bitova potreban za kodiranje klase primera iz skupa S, gde su češćim slučajevima kompresijom dodeljeni kraći kodovi..
- Za problem gde postoji više klasa ishoda (multi-class), entropija se računa kao:

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

gde je c broj klasa.

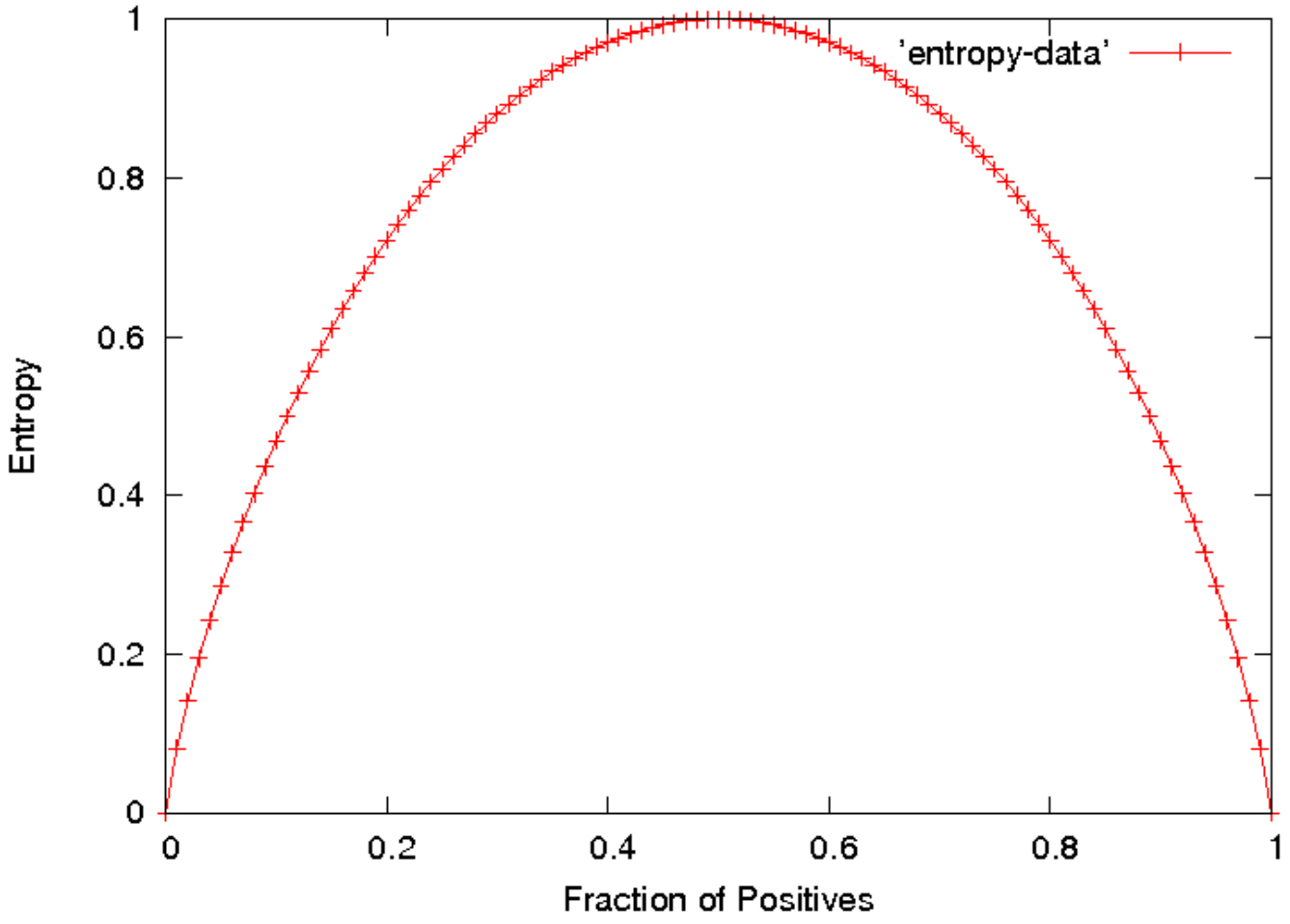
Entropija

- Ako ima n jednako verovatnih poruka, onda je verovatnoća p svake poruke 1/n, a informacija koja se prenosi porukom je -log(p) = log(n). *[osnova logaritma je 2.]*
- To znači, ako ima 16 poruka, onda je log(16) = 4, i treba nam 4 bita
- Uopšteno, ako nam je data raspodela verovatnoća $P = (p_1, p_2, .., p_n)$, onda se *informacija koju nosi ova disribucija*, zove *entropija* za *P*:
  - $I(P) = -(p_1*\log(p_1) + p_2*\log(p_2) + .. + p_n*\log(p_n))$.
- Na primer: ako je P (0.5, 0.5) onda je I(P) = 1, ako je P (0.67, 0.33) onda je I(P) = 0.92, ako je P (1, 0) onda je I(P) = 0. (Što je uniformnija raspodela, više informacija nosi)

Entropija

- Entropija bacanja fer novčića je jedan bit po bacanju.
- Međutim, ako novčić nije der, onda je neizvesnot, a samim tim i entropija, manja.
- To je zato što, ako nas neko pita da predvidimo ishod bacanja, možemo birati klasu (rezultat) koji je češći i biti u pravu veći broj puta nego u krivu.
- Informacija da je novčić ne fer, znači da nam treba manje od jednog bita po bacanju da bismo preneli poruku.

Graf entropije za binarnu klasifikaciju



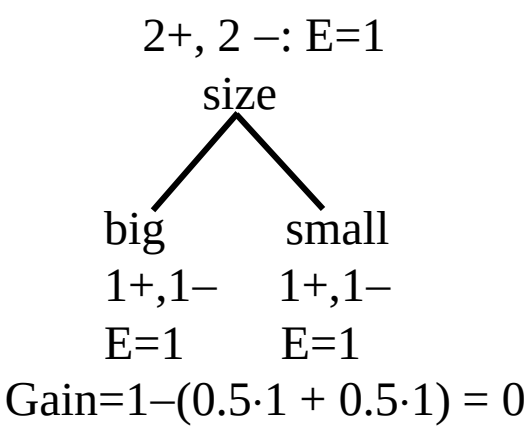
Information Gain

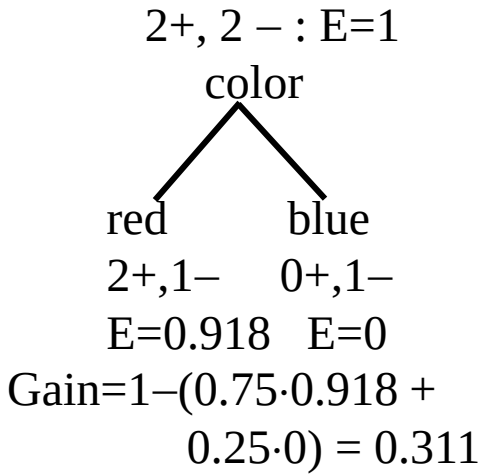
- Information gain atributa F je očekivano smanjenje u entropiji koje proizilazi iz podele po ovom atributu.

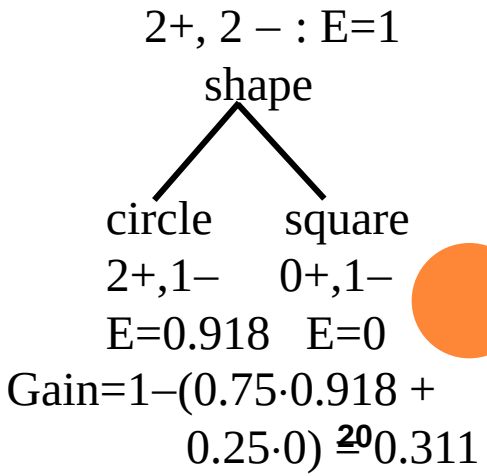
$$Gain(S,F) = Entropy(S) - \sum_{v \in Values(F)} \frac{|S_v|}{|S|} Entropy(S_v)$$

gde je S_v podskup skupa S koji ima vrednost v za atribut F.

- Drugi sabirak je entropija svakog rezultujućeg podskupa pomnožena svojom relativnom veličinom.
- Primer:
  - <big, red, circle>: + <small, red, circle>: +
  - <small, red, square>: – <big, blue, circle>: –







Primer 1

Kreirati stablo odlučivanja za određivanje da li životinja leže jaja. (eng. lays eggs)

Independent/Condition attributes					Dependent/ Decision attributes
Animal	**Warm-blooded**	**Feathers**	**Fur**	**Swims**	**Lays Eggs**
Ostrich	Yes	Yes	No	No	Yes
Crocodile	No	No	No	Yes	Yes
Raven	Yes	Yes	No	No	Yes
Albatross	Yes	Yes	No	No	Yes
Dolphin	Yes	No	No	Yes	No
Koala	Yes	No	Yes	No	No

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

$$Gain(S, F) = Entropy(S) - \sum_{v \in Values(F)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Entropy(4Y,2N) = $-(4/6)\log_2(4/6) - (2/6)\log_2(2/6)$

= 0.91829

- Sada je potrebno pronaći Information Gain za sve atribute: Warm-blooded, Feathers, Fur, Swims.

$$Gain(S,F) = Entropy(S) - \sum_{v \in Values(F)} \frac{|S_v|}{|S|} Entropy(S_v)$$

- Za atribut ‘Warm-blooded’:

Vrednosti(Warm-blooded) : [Yes,No]

S = [4Y,2N]

S_{Yes} = [3Y,2N] E(S_{Yes}) = 0.97095

S_{No} = [1Y,0N] E(S_{No}) = 0 (svi članovi pripadaju istoj klasi)

Gain(S,Warm-blooded) = 0.91829 – [(5/6)*0.97095 + (1/6)*0]
= 0.10916

- Za atribut ‘Feathers’:

Vrednosti(Feathers) : [Yes,No]

S = [4Y,2N]

S_{Yes} = [3Y,0N] E(S_{Yes}) = 0

S_{No} = [1Y,2N] E(S_{No}) = 0.91829

Gain(S,Feathers) = 0.91829 – [(3/6)*0 + (3/6)*0.91829]
= 0.45914

$$Gain(S,F) = Entropy(S) - \sum_{v \in Values(F)} \frac{|S_v|}{|S|} Entropy(S_v)$$

- Za atribut ‘Fur’:

Vrednosti(Fur) : [Yes,No]

S = [4Y,2N]

S_{Yes} = [0Y,1N] E(S_{Yes}) = 0

S_{No} = [4Y,1N] E(S_{No}) = 0.7219

Gain(S,Fur) = 0.91829 – [(1/6)*0 + (5/6)*0.7219]

= 0.3167

- Za atribut ‘Swims’:

Vrednosti(Swims) : [Yes,No]

S = [4Y,2N]

S_{Yes} = [1Y,1N] E(S_{Yes}) = 1 (isti broj članova obe klase)

S_{No} = [3Y,1N] E(S_{No}) = 0.81127

Gain(S,Swims) = 0.91829 – [(2/6)*1 + (4/6)*0.81127]

= 0.04411

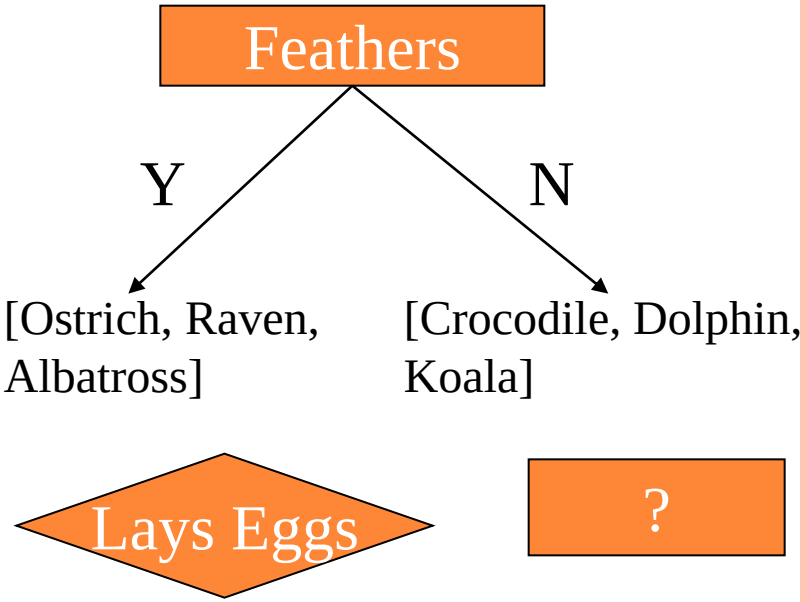
Gain(S,Warm-blooded) = 0.10916; Gain(S,Feathers) = 0.45914

Gain(S,Fur) = 0.31670; Gain(S,Swims) = 0.04411

Gain(S,Feathers) je maksimum, pa je izabran za koreni čvor.

Animal	Warm-blooded	Feathers	Fur	Swims	Lays Eggs
Ostrich	Yes	Yes	No	No	Yes
Crocodile	No	No	No	Yes	Yes
Raven	Yes	Yes	No	No	Yes
Albatross	Yes	Yes	No	No	Yes
Dolphin	Yes	No	No	Yes	No
Koala	Yes	No	Yes	No	No

'Y' naslednik ima samo pozitivne primere i postaje čvor list sa klasom 'Lays Eggs'



Animal	Warm-blooded	Feathers	Fur	Swims	Lays Eggs
Crocodile	No	No	No	Yes	Yes
Dolphin	Yes	No	No	Yes	No
Koala	Yes	No	Yes	No	No

- Sada ponavljamo postupak:

S: [Crocodile, Dolphin, Koala]

S: [1+,2-]

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

Entropy(S) = $-(1/3)\log_2(1/3) - (2/3)\log_2(2/3)$

= 0.91829

- Za atribut ‘Warm-blooded’:

Vrednosti(Warm-blooded) : [Yes,No]
S = [1Y,2N]
S_{Yes} = [0Y,2N] $E(S_{Yes}) = 0$
S_{No} = [1Y,0N] $E(S_{No}) = 0$
Gain(S,Warm-blooded) = 0.91829 – [(2/3)*0 + (1/3)*0] = **0.91829**

- Za atribut ‘Fur’:

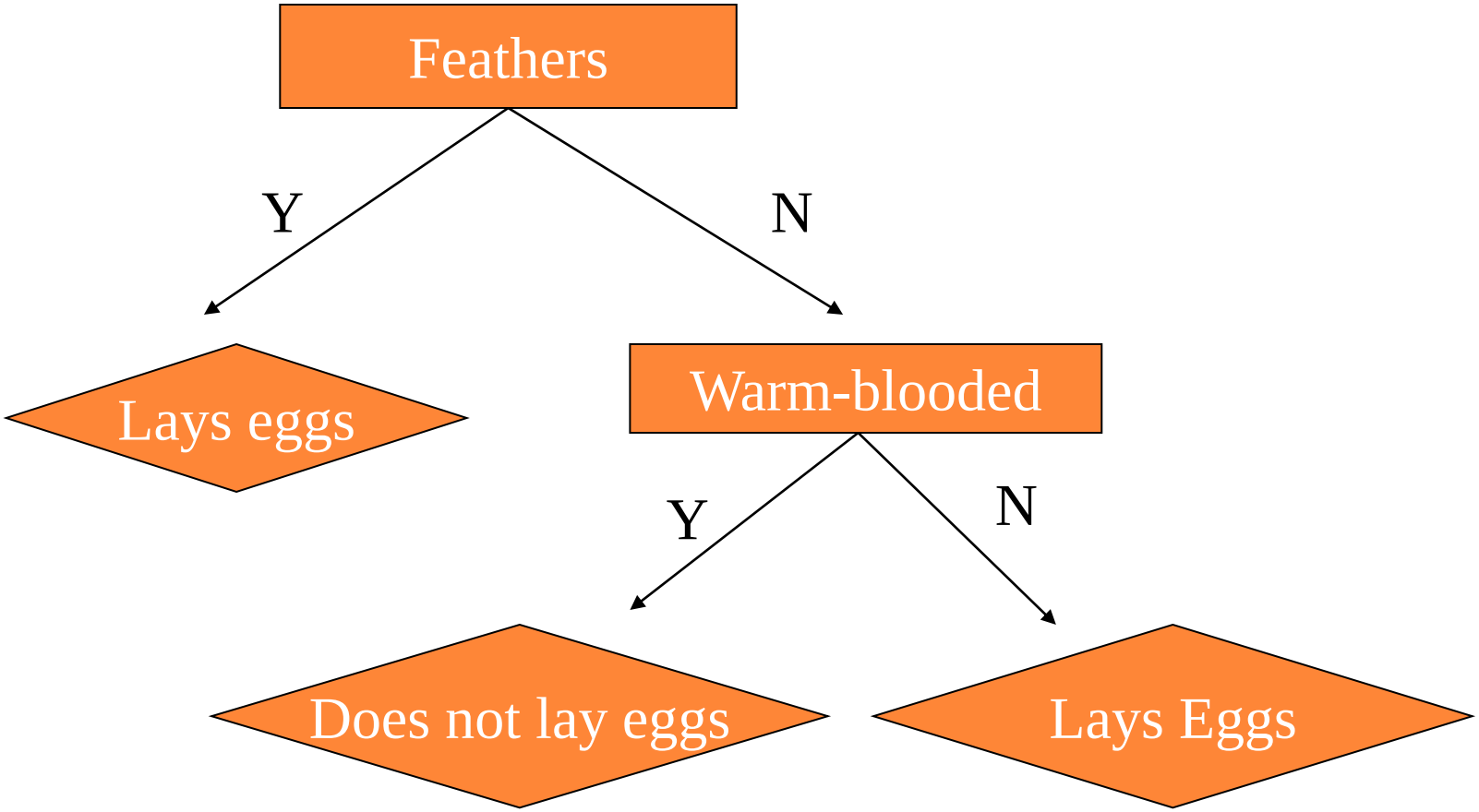
Vrednosti(Fur) : [Yes,No]
S = [1Y,2N]
S_{Yes} = [0Y,1N] $E(S_{Yes}) = 0$
S_{No} = [1Y,1N] $E(S_{No}) = 1$
Gain(S,Fur) = 0.91829 – [(1/3)*0 + (2/3)*1] = **0.25162**

- Za atribut ‘Swims’:

Vrednosti(Swims) : [Yes,No]
S = [1Y,2N]
S_{Yes} = [1Y,1N] $E(S_{Yes}) = 1$
S_{No} = [0Y,1N] $E(S_{No}) = 0$
Gain(S,Swims) = 0.91829 – [(2/3)*1 + (1/3)*0] = **0.25162**

Gain(S,Warm-blooded) je maksimum.

Konačno stablo odlučivanja će biti:



Primer 2

Kreirati stablo na osnovu kojega će se odrediti da li će neko izgoreti na suncu (eng. sunburn).

Name	Hair	Height	Weight	Lotion	Sunburned
Sarah	Blonde	Average	Light	No	Yes
Dana	Blonde	Tall	Average	Yes	No
Alex	Brown	Short	Average	Yes	No
Annie	Blonde	Short	Average	No	Yes
Emily	Red	Average	Heavy	No	Yes
Pete	Brown	Tall	Heavy	No	No
John	Brown	Average	Heavy	No	No
Katie	Blonde	Short	Light	Yes	No

- S = [3+, 5-]

Entropy(S) = $-(3/8)\log_2(3/8) - (5/8)\log_2(5/8)$

= 0.95443

Pronaći IG za sva 4 atributa: Hair, Height, Weight, Lotion.

- Za atribut ‘Hair’:

Vrednosti(Hair) : [Blonde, Brown, Red]

S = [3+,5-]

S_{Blonde} = [2+,2-] $E(S_{Blonde}) = 1$

S_{Brown} = [0+,3-] $E(S_{Brown}) = 0$

S_{Red} = [1+,0-] $E(S_{Red}) = 0$

Gain(S,Hair) = 0.95443 – [(4/8)*1 + (3/8)*0 + (1/8)*0]

= 0.45443

- Za atribut ‘Height’:

Vrednosti(Height) : [Average, Tall, Short]

$S_{Average}$ = [2+,1-] $E(S_{Average}) = 0.91829$

S_{Tall} = [0+,2-] $E(S_{Tall}) = 0$

S_{Short} = [1+,2-] $E(S_{Short}) = 0.91829$

Gain(S,Height) = 0.95443 – [(3/8)*0.91829 + (2/8)*0 + (3/8)*0.91829]

= 0.26571

- Za atribut ‘Weight’:

Vrednosti(Weight) : [Light, Average, Heavy]

S_{Light} = [1+,1-] $E(S_{Light}) = 1$

$S_{Average}$ = [1+,2-] $E(S_{Average}) = 0.91829$

S_{Heavy} = [1+,2-] $E(S_{Heavy}) = 0.91829$

Gain(S,Weight) = 0.95443 – [(2/8)*1 + (3/8)*0.91829 + (3/8)*0.91829]

= 0.01571

- Za atribut ‘Lotion’:

Vrednosti(Lotion) : [Yes, No]

SYes = [0+,3-] $E(S_{Yes}) = 0$

S_{No} = [3+,2-] $E(S_{No}) = 0.97095$

Gain(S,Lotion) = 0.95443 – [(3/8)*0 + (5/8)*0.97095]

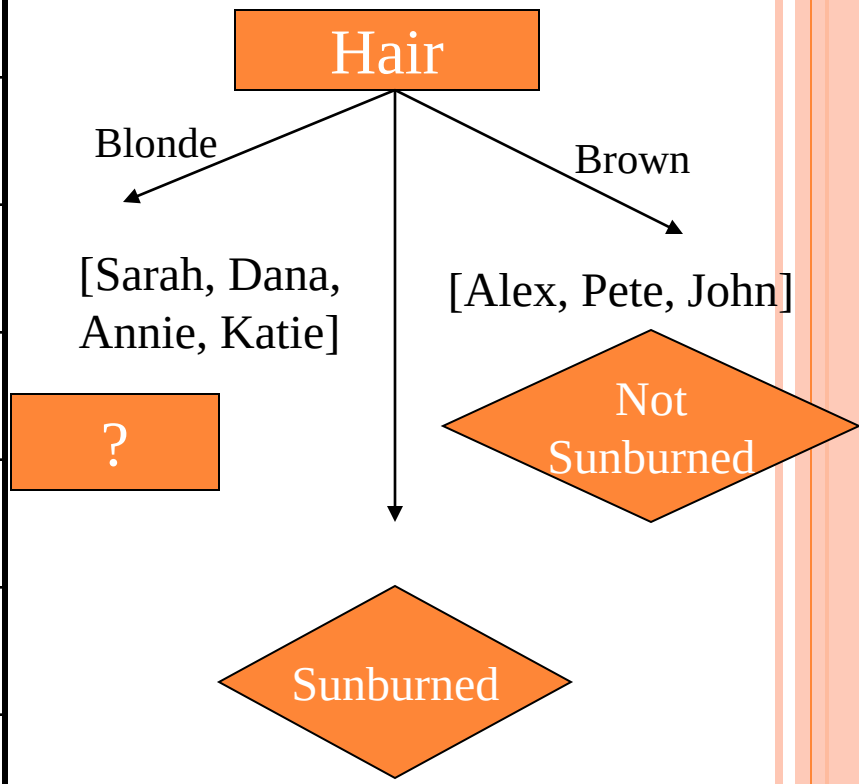
= 0.3475

Gain(S,Hair) = 0.45443; Gain(S,Height) = 0.26571

Gain(S,Weight) = 0.01571; Gain(S,Lotion) = 0.3475

Gain(S,Hair) je maksimum, pa je izabrani da bude koreni čvor.

Name	Hair	Height	Weight	Lotion	Sunburned
Sarah	Blonde	Average	Light	No	Yes
Dana	Blonde	Tall	Average	Yes	No
Alex	Brown	Short	Average	Yes	No
Annie	Blonde	Short	Average	No	Yes
Emily	Red	Average	Heavy	No	Yes
Pete	Brown	Tall	Heavy	No	No
John	Brown	Average	Heavy	No	No
Katie	Blonde	Short	Light	Yes	No



Name	Hair	Height	Weight	Lotion	Sunburned
Sarah	Blonde	Average	Light	No	Yes
Dana	Blonde	Tall	Average	Yes	No
Annie	Blonde	Short	Average	No	Yes
Katie	Blonde	Short	Light	Yes	No

Ponavljamo postupak:

S = [Sarah, Dana, Annie, Katie]
S: [2+,2-]
Entropy(S) = 1

Pronaći IG za preostala 3 atributa: Height, Weight, Lotion

- Za atribut ‘Height’:

Vrednosti(Height) : [Average, Tall, Short]
S = [2+,2-]
$S_{Average}$ = [1+,0-] $E(S_{Average}) = 0$
S_{Tall} = [0+,1-] $E(S_{Tall}) = 0$
S_{Short} = [1+,1-] $E(S_{Short}) = 1$
Gain(S,Height) = 1 – [(1/4)*0 + (1/4)*0 + (2/4)*1]
= 0.5

- Za atribut ‘Weight’:

Vrednosti(Weight) : [Average, Light]

S = [2+,2-]

$S_{Average}$ = [1+,1-] $E(S_{Average}) = 1$

S_{Light} = [1+,1-] $E(S_{Light}) = 1$

Gain(S,Weight) = 1 – [(2/4)*1 + (2/4)*1]

= 0

- Za atribut ‘Lotion’:

Vrednosti(Lotion) : [Yes, No]

S = [2+,2-]

S_{Yes} = [0+,2-] $E(S_{Yes}) = 0$

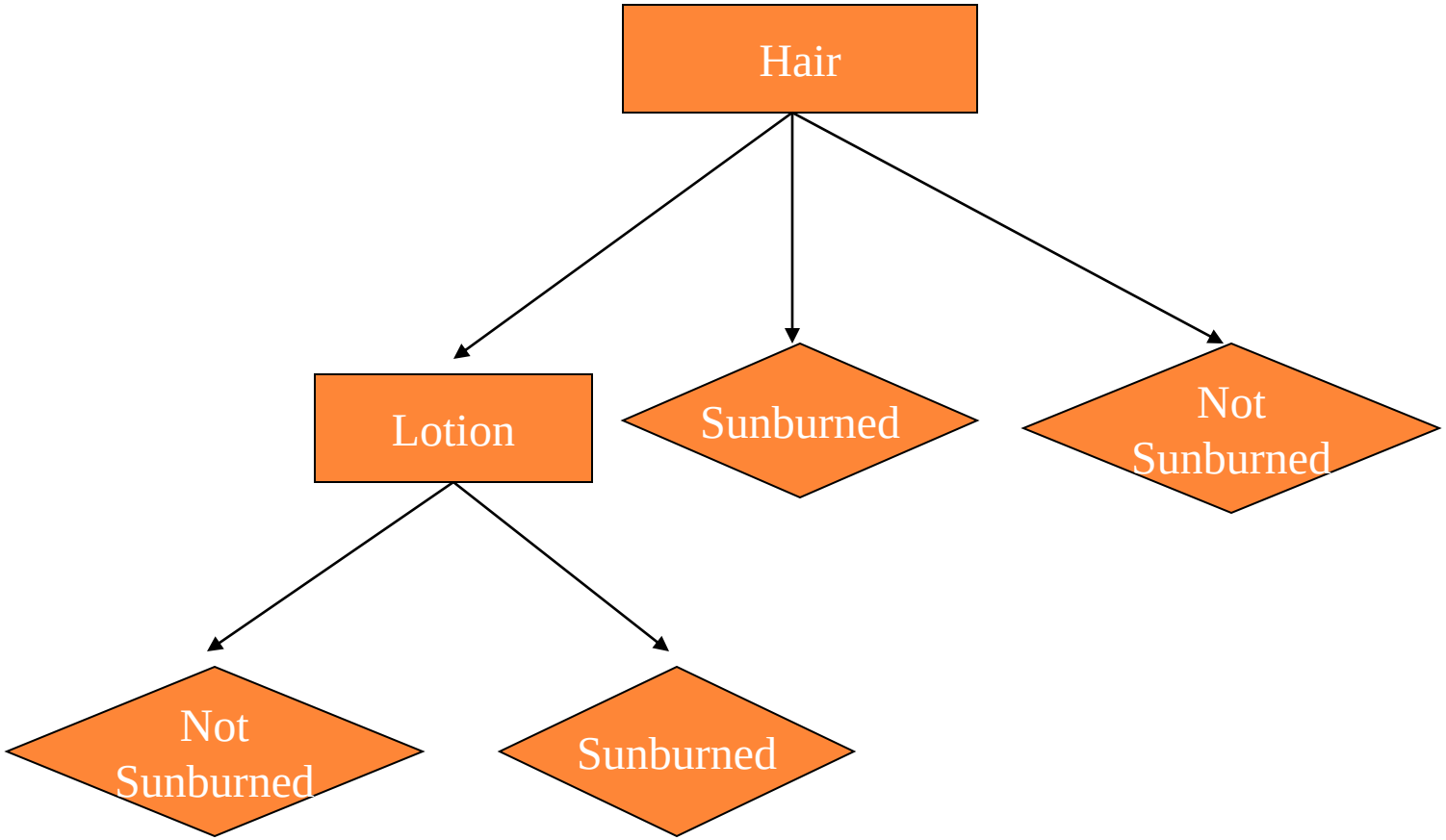
S_{No} = [2+,0-] $E(S_{No}) = 0$

Gain(S,Lotion) = 1 – [(2/4)*0 + (2/4)*0]

= 1

Gain(S,Lotion) je maksimum.

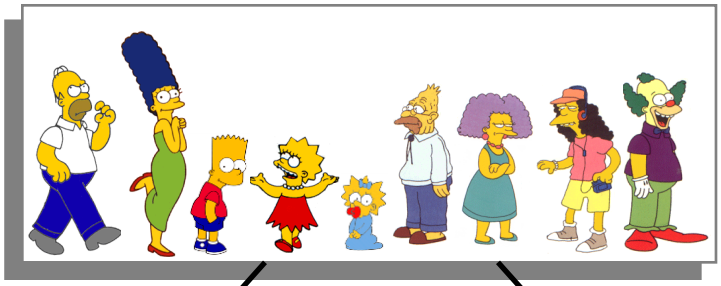
- Konačno stablo odlučivanja će biti:



Primer 3: Simpsonovi

Person	Hair Length	Weight	Age	Class
 Homer	0″	250	36	**M**
 Marge	10″	150	34	**F**
 Bart	2″	90	10	**M**
 Lisa	6″	78	8	**F**
 Maggie	4″	20	1	**F**
 Abe	1″	170	70	**M**
 Selma	8″	160	41	**F**
 Otto	10″	180	38	**M**
 Krusty	6″	200	45	**M**

 Comic	8″	290	38	**?**

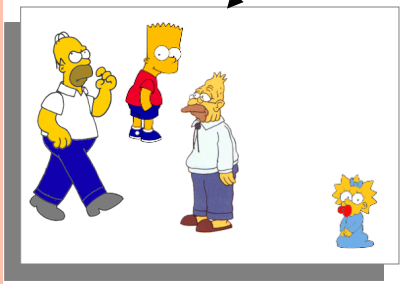


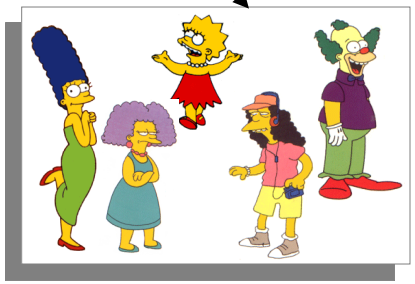
$$Entropy(S) = -p_1 \log_2(p_1) \quad - \quad p_2 \log_2(p_2)$$

$Entropy(4\mathbf{F},5\mathbf{M}) = -(4/9)\log_2(4/9) - (5/9)\log_2(5/9)$
$= \mathbf{0.9911}$

yes

no



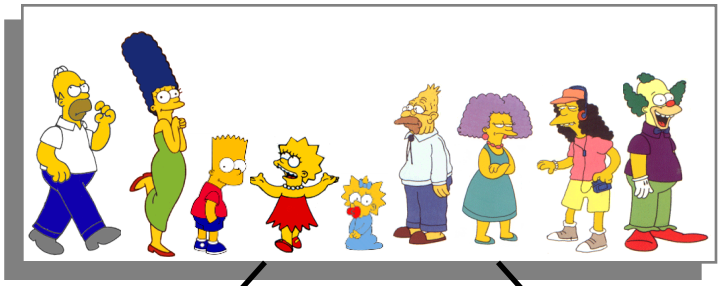


Pokušajmo da podelimo po: *Hair length*

$Entropy(1\mathbf{F},3\mathbf{M}) = -(1/4)\log_2(1/4) - (3/4)\log_2(3/4)$
$= \mathbf{0.8113}$

$Entropy(3\mathbf{F},2\mathbf{M}) = -(3/5)\log_2(3/5) - (2/5)\log_2(2/5)$
$= \mathbf{0.9710}$

Gain(Hair Length <= 5) = **0.9911** – (4/9 * **0.8113** + 5/9 * **0.9710**) = **0.0911**

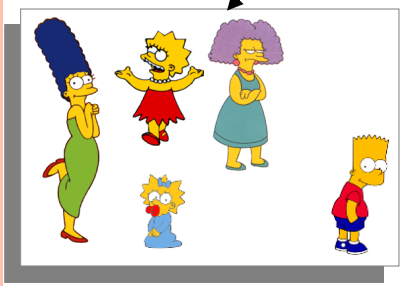


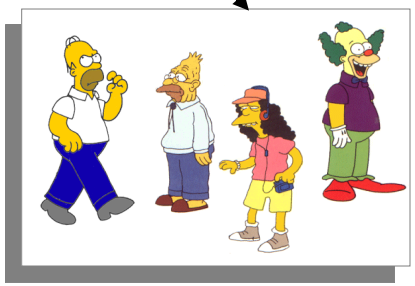
$$Entropy(S) = -p_1 \log_2(p_1) \quad - \quad p_2 \log_2(p_2)$$

$Entropy(4\mathbf{F},5\mathbf{M}) = -(4/9)\log_2(4/9) - (5/9)\log_2(5/9)$
$= \mathbf{0.9911}$

yes

no



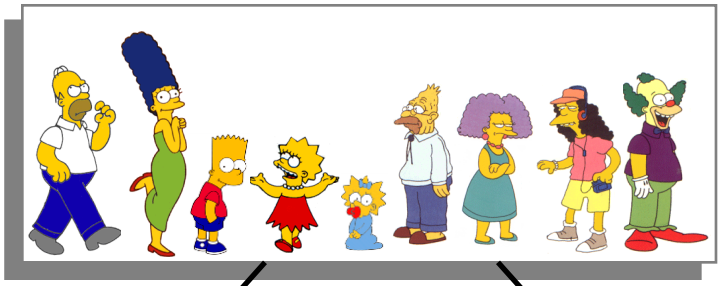


Pokušajmo da podelimo po: *Weight*

$Entropy(4\mathbf{F},1\mathbf{M}) = -(4/5)\log_2(4/5) - (1/5)\log_2(1/5)$
$= \mathbf{0.7219}$

$Entropy(0\mathbf{F},4\mathbf{M}) = -(0/4)\log_2(0/4) - (4/4)\log_2(4/4)$
$= \mathbf{0}$

$Gain(\text{Weight} <= 160) = \mathbf{0.9911} - (5/9 * \mathbf{0.7219} + 4/9 * \mathbf{0}) = \mathbf{0.5900}$

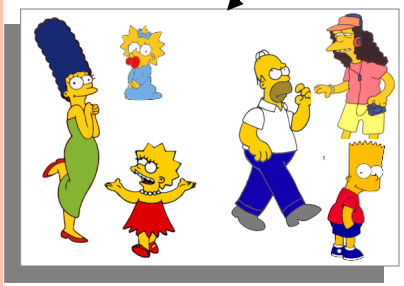


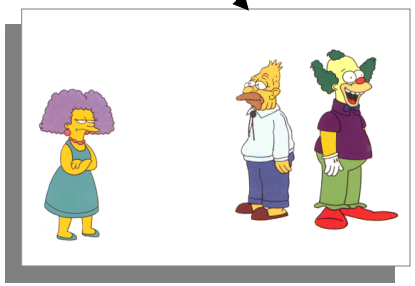
$$Entropy(S) = -p_1 \log_2(p_1) \quad - \quad p_2 \log_2(p_2)$$

$Entropy(4\mathbf{F},5\mathbf{M}) = -(4/9)\log_2(4/9) - (5/9)\log_2(5/9)$
$= \mathbf{0.9911}$

yes

no





Pokušajmo da podelimo po: *Age*

$Entropy(3\mathbf{F},3\mathbf{M}) = -(3/6)\log_2(3/6) - (3/6)\log_2(3/6)$
$= \mathbf{1}$

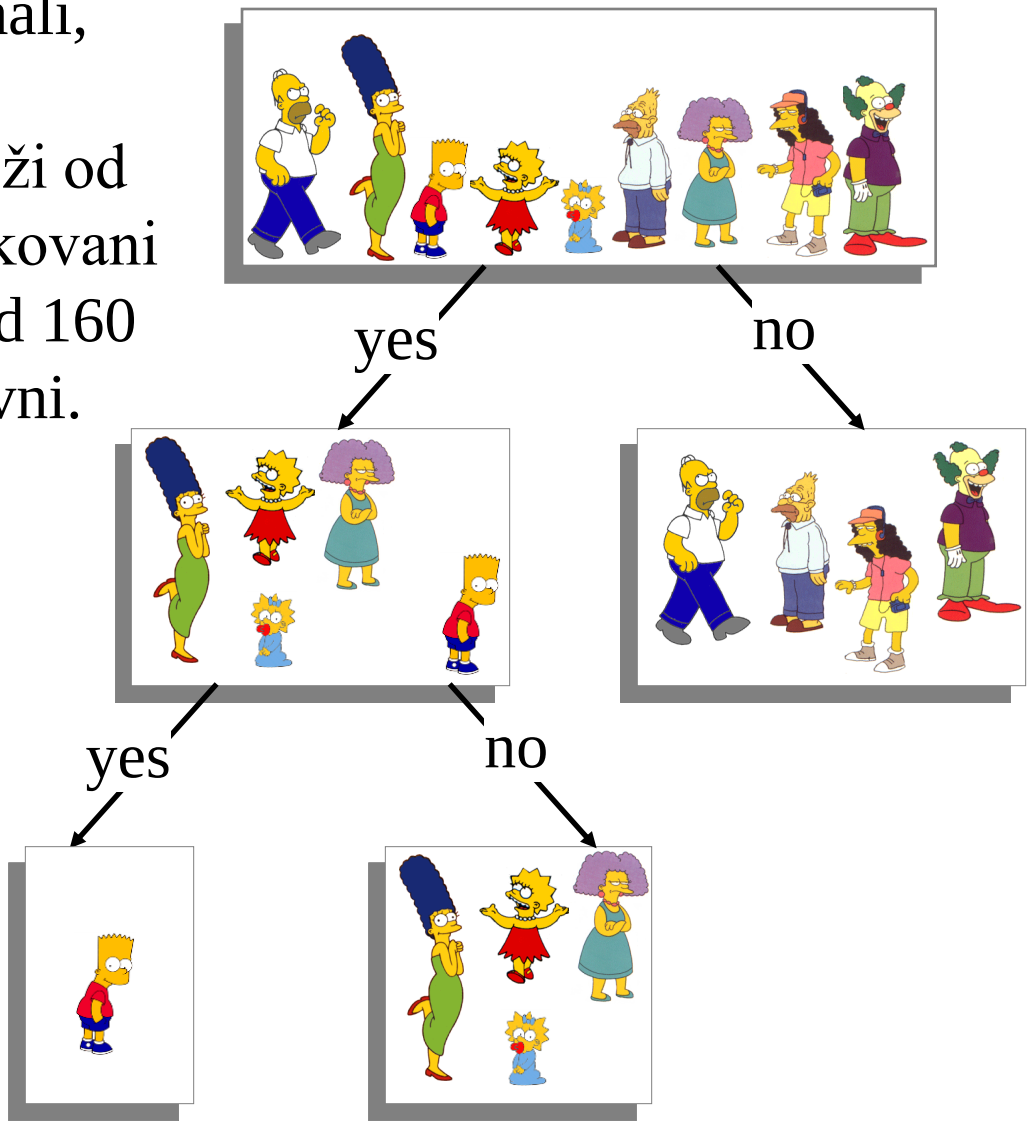
$Entropy(1\mathbf{F},2\mathbf{M}) = -(1/3)\log_2(1/3) - (2/3)\log_2(2/3)$
$= \mathbf{0.9183}$

$Gain(\text{Age} <= 40) = \mathbf{0.9911} - (6/9 * \mathbf{1} + 3/9 * \mathbf{0.9183}) = \mathbf{0.0183}$

- Od 3 atributa koje smo imali, *Weight* je bio najbolji.
- Ali, dok su ljudi koji su teži od 160 funti savršeno klasifikovani (kao muškarci), ljudi ispod 160 funti nisu dobro klasifikovni.

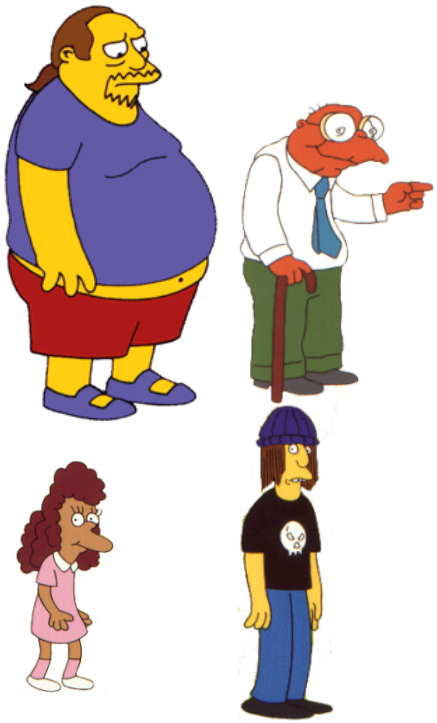
- Nastavljamo proceduru.

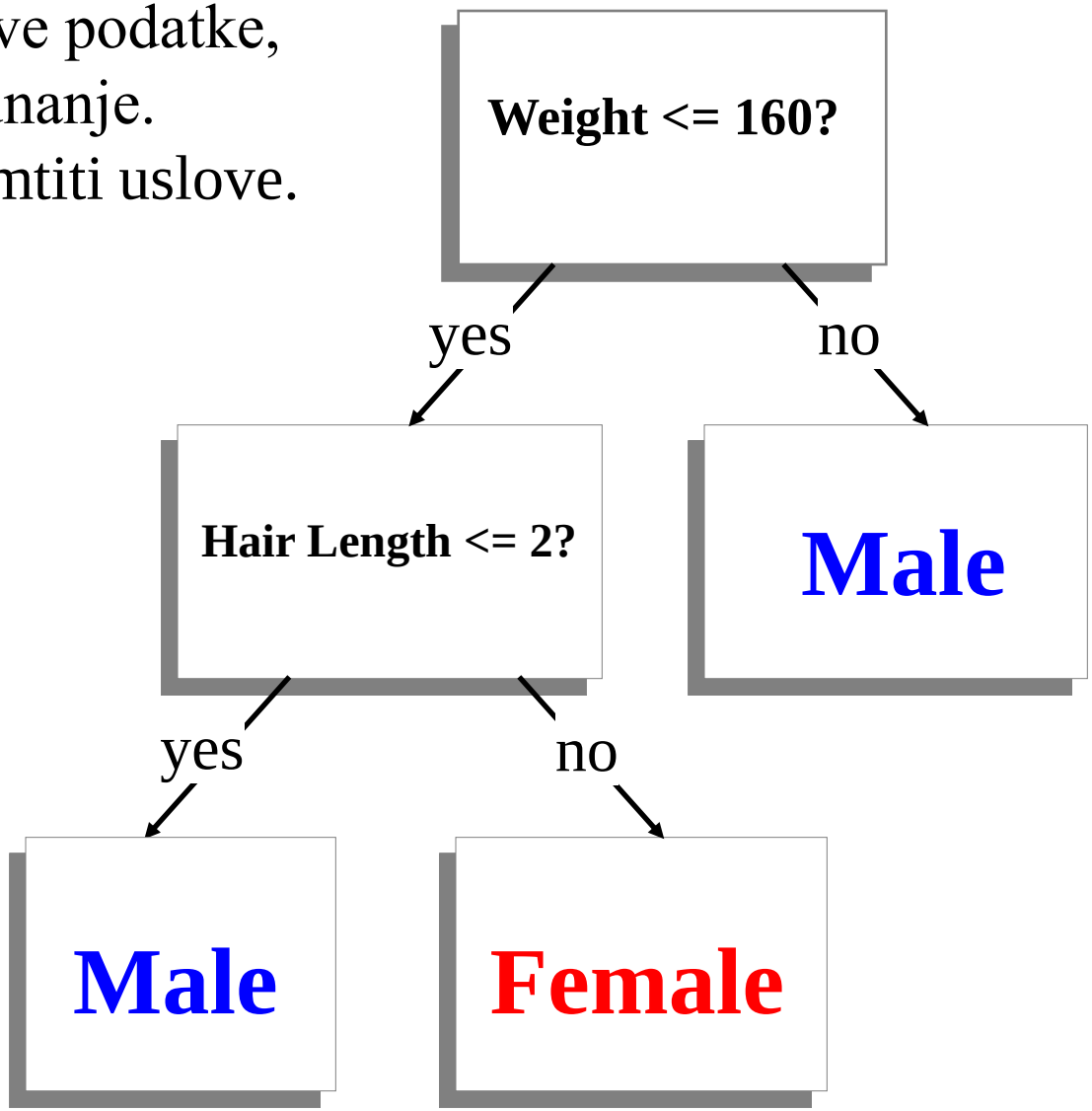
- Ovog puta podelu možemo izvršiti po atributu *Hair length*, i dolazimo do kraja!



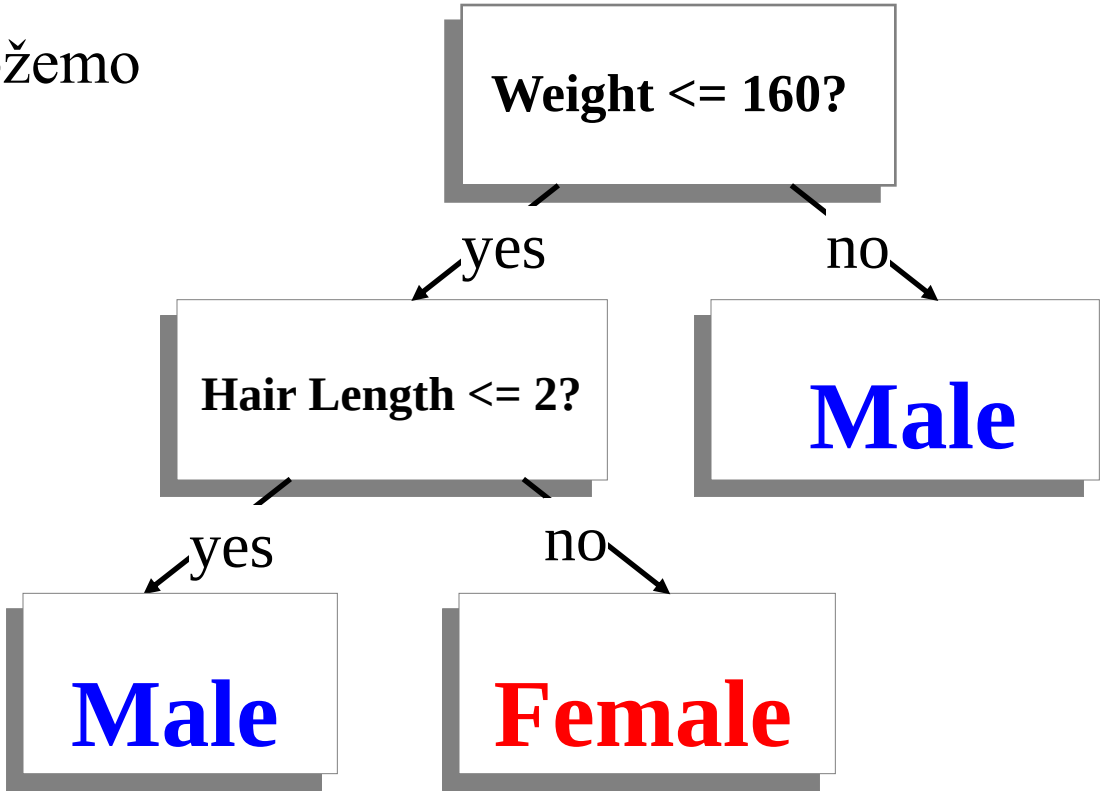
- Nije potrebo čuvati sve podatke, niti iznova raditi računanje.
- Jedino moramo zapamtiti uslove.

- Kako bi ovi ljudi bili klasifikovani?





- Vrlo jednostavno možemo napraviti pravila od stabla odlučivanja.

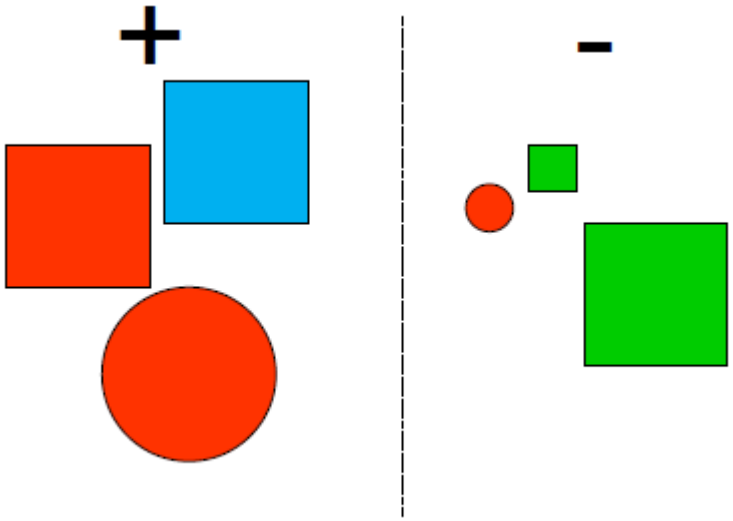


Pravila za klasifikaciju Males/Females

If *Weight* **veće od** 160, klasifikuj kao **Male**
Elseif *Hair Length* **manje od ili jednako** to 2,
klasifikuj kao **Male**
Else klasifikuj kao **Female**

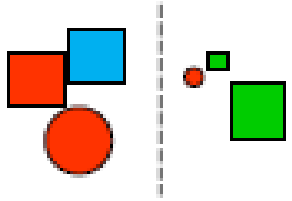
Primer 4

- Atributi: color, shape, size
- Koji je najbolji čvor za koren stabla?



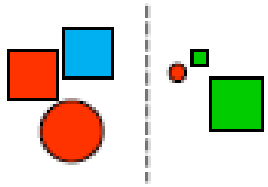
TRENING SKUP

Example	Color	Shape	Size	Class
1	Red	Square	Big	+
2	Blue	Square	Big	+
3	Red	Circle	Big	+
4	Red	Circle	Small	-
5	Green	Square	Small	-
6	Green	Square	Big	-



H(class)=
H(class | color)=

Example	Color	Shape	Size	Class
1	Red	Square	Big	+
2	Blue	Square	Big	+
3	Red	Circle	Big	+
4	Red	Circle	Small	-
5	Green	Square	Small	-
6	Green	Square	Big	-



H(class)= H(3/6,3/6) = 1

H(class | color)= 3/6 * H(2/3,1/3) + 1/6 * H(1,0) + 2/6 * H(0,1)

3 out of 6 are red

2 of the red are +

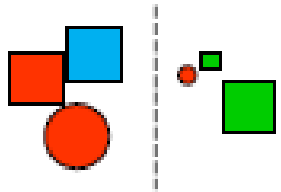
1 out of 6 is blue

blue is +

2 out of 6 are green

green is -

Example	Color	Shape	Size	Class
1	Red	Square	Big	+
2	Blue	Square	Big	+
3	Red	Circle	Big	+
4	Red	Circle	Small	-
5	Green	Square	Small	-
6	Green	Square	Big	-

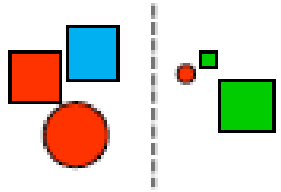


H(class)= H(3/6,3/6) = 1

H(class | color)= 3/6 * H(2/3,1/3) + 1/6 * H(1,0) + 2/6 * H(0,1)

I(class; color) = H(class) – H(class | color) = 0.54 bits

Example	Color	Shape	Size	Class
1	Red	Square	Big	+
2	Blue	Square	Big	+
3	Red	Circle	Big	+
4	Red	Circle	Small	-
5	Green	Square	Small	-
6	Green	Square	Big	-



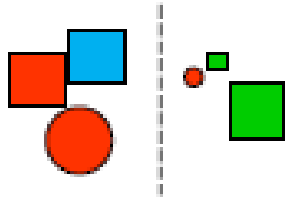
H(class)= H(3/6,3/6) = 1

H(class | shape)= 4/6 * H(1/2, 1/2) + 2/6 * H(1/2,1/2)

I(class; shape) = H(class) – H(class | shape) = 0 bits

Shape tells us nothing about the class!

Example	Color	Shape	Size	Class
1	Red	Square	Big	+
2	Blue	Square	Big	+
3	Red	Circle	Big	+
4	Red	Circle	Small	-
5	Green	Square	Small	-
6	Green	Square	Big	-

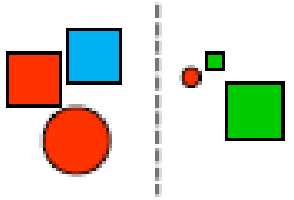


H(class)= H(3/6,3/6) = 1

H(class | size)= 4/6 * H(3/4, 1/4) + 2/6 * H(0,1)

I(class; size) = H(class) – H(class | size) = 0.46 bits

Example	Color	Shape	Size	Class
1	Red	Square	Big	+
2	Blue	Square	Big	+
3	Red	Circle	Big	+
4	Red	Circle	Small	-
5	Green	Square	Small	-
6	Green	Square	Big	-

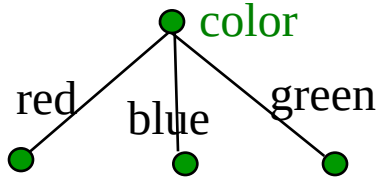


I(class; color) = H(class) – H(class | color) = 0.54 bits
I(class; shape) = H(class) – H(class | shape) = 0 bits
I(class; size) = H(class) – H(class | size) = 0.46 bits

- IZABRATI COLOR KAO NAJBOLJI ATRIBUT ZA KORENI ČVOR.

<big, red, circle>: +
<small, red, square>: –

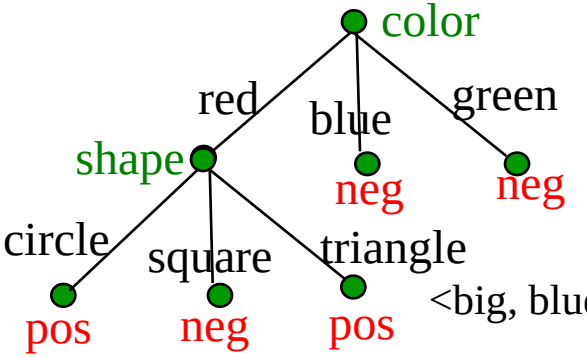
<small, red, circle>: +
<big, blue, circle>: –



<big, red, circle>: +
<small, red, circle>: +
<small, red, square>: –

<big, red, circle>: +
<small, red, square>: –

<small, red, circle>: +
<big, blue, circle>: –



<big, red, circle>: +
<small, red, circle>: +
<small, red, square>: –

<big, blue, circle>: –

<big, red, circle>: +
<small, red, circle>: +

<small, red, square>: –