Niech $f : \{0,1\}^3 \mapsto \{0,1\}$ będzie funkcją boolowską zdefiniowaną jako $f(x,y,z) = 1$, wtedy i tylko wtedy gdy co najmniej dwie zmienne x, y, z wynoszą 1. Znajdź DNF i CNF.

Tworzymy tabelkę jeśli „1” w wierszu występuję więcej niż 2 razy to f(x,y,z) równa się 1

DNF (koniunkcja w nawiasach) dla „1”

CNF (alternatywa w nawiasiach) dla „0” i zmieniamy znak

Czy to zdanie $(\alpha \wedge \beta) \Rightarrow \gamma$ jest spełnialne? Znajdź DNF dla tej funkcji boolowskiej

Zdanie spełnialne to takie, które NIE jest tautologią (czyli musi mieć jakieś 0, jak same 1 to tautologia)

Trzeba pamiętać, że implikacja nie może występować dla DNF i CNF więc implikacja to inaczej zanegowanie całej lewej strony i zmiana znaku na alternatywę czyli ~(a&b)vc potem prawo De Morgana ~a v ~b v c to jest poprawna wersja dla DNF, teraz trzeba stworzyć tabelkę i pokazać DNF

Dla poniższego systemu decyzyjnego, jaką decyzję należy przypisać do nowego obiektu testowego t' używając modelu 3-NN z odległością Hamminga?

	a_1	a_2	a_3	dec
o_1	1	1	0	0
o_2	0	0	1	0
o_3	0	0	1	0
o_4	1	1	1	1
o_5	1	1	0	1
t'	0	1	1	?

3NN oznacza, że musimy znaleźć trzy decyzje. Tutaj wystarczy spojrzeć na t’ i porównywać wiersze o czyli o1 posiada 110 a t’ 011 więc różnica wynosi 2 ponieważ o(1)->t’(0) i o(0)->t’(1) te wartości się różnią. Szukamy najmniejszych wartości po czym patrzymy na „dec”, jeśli więcej jest 0 to odpowiedź to 0 jeśli 1 to odpowiedź 1 za znak zapytania.

Dla poniższego systemu decyzyjnego, znajdź redukt decyzyjny. Znajdź wszystkie wygenerowane z niego reguły (w ich możliwie najkrótszej formie).

	a_1	a_2	a_3	dec
o_1	1	1	0	1
o_2	1	0	1	0
o_3	0	1	1	1
o_4	1	1	1	0
o_5	1	1	0	1

Tutaj trzeba patrzeć na zależności w kolumnach a i na dec, spójrzmy na a1a2 1,1->0 oraz 1,1->0 oraz 1,1->1 więc jest sprzeczność, spójrzmy na a1a3 1,0->1 oraz 1,1->0 oraz 0,1->0 wszystko się zgadza.

Piszemy teraz w taki sposób:

a1=1 & a2=0 -> dec = 1

a1=1 & a2=1 -> dec = 0

a1=0 & a2=1 -> dec = 1

Patrzymy na wartości unikalne w a1 i a2 jak w kolumnie widzimy, że jest jedno 0 tak samo w a2 więc można to skrócić

a1=1 & a2=1 -> dec = 0

a1=0 & a2=0 -> dec = 1

To jest nasza odpowiedź

Skonstruuj perceptron odpowiadający funkcji boolowskiej $\neg X \vee (X \wedge \neg Y)$. Zaleca się zapisanie zależności między parametrami wagi w celu znalezienia perceptronu.

Perceptron 1: ¬X

Wagi: W1 = -1 (dla X)

Bias: b1 = 0.5

Wyjście: ¬X = Heaviside(W1*X + b1)

Perceptron 2: X ∧ ¬Y

Wagi: W2 = 1 (dla X), W3 = -1 (dla Y)

Bias: b2 = 1.5

Wyjście: X ∧ ¬Y = Heaviside(W2X + W3Y + b2)

Perceptron 3: ¬X ∨ (X ∧ ¬Y)

Wagi: W4 = 1 (dla Perceptron 1), W5 = 1 (dla Perceptron 2)

Bias: b3 = -0.5

Wyjście: ¬X ∨ (X ∧ ¬Y) = Heaviside(W4¬X + W5(X ∧ ¬Y) + b3)

Dla perceptronu w warstwie 2, bierze on na wejściu wyniki perceptronów z warstwy 1 i wykonuje na nich operację OR.

Na przykład, jeżeli X=0 i Y=0, to:

¬X = 1 (z Perceptron 1)

X ∧ ¬Y = 0 (z Perceptron 2)

Zatem, ¬X ∨ (X ∧ ¬Y) = 1, co jest wynikiem końcowym dla tych wartości wejściowych.

NOT to W0=-1 b=0.5

AND to W1=1, W2=1, b=-1.5

OR to W1=1, W2=1, b=-0.5

Skonstruuj perceptron odpowiadający funkcji boolowskiej $\neg X \wedge (X \vee \neg Y)$. Zaleca się zapisanie zależności między parametrami wagi w celu znalezienia perceptronu.

Budowa pierwszej warstwy sieci neuronowej

A. Perceptron dla ¬X:

Funkcja NOT dla X będzie miał wagę -1 dla X i bias równy 0.5. Dlatego, gdy X=1, output będzie 0, a gdy X=0, output będzie 1.

B. Perceptron dla (X ∨ ¬Y):

Funkcja OR między X i ¬Y będzie miał wagę 1 dla X, wagę -1 dla Y (realizując funkcję NOT dla Y), i bias -0.5. Dlatego output będzie 1, gdy X=1 lub Y=0.

Budowa drugiej warstwy sieci neuronowej

A. Perceptron dla ¬X ∧ (X ∨ ¬Y) :

Perceptron realizujący funkcję AND będzie miał wagi równe 1 dla obu wejść i bias -1.5.

Output będzie 1 tylko wtedy, gdy oba wejścia są równe 1.

Aproksymacja Górna oraz Dolna

	a_1	a_2	a_3	d
o_1	wysoka	bliski	średni	tak
o_2	wysoka	bliski	średni	tak
o_3	wysoka	bliski	średni	tak
o_4	więcej niż średnia	daleki	silny	nie pewne
o_5	więcej niż średnia	daleki	silny	nie
o_6	więcej niż średnia	daleki	lekki	nie
o_7	wysoka	bliski	średni	tak
o_8	więcej niż średnia	daleki	lekki	nie
o_9	więcej niż średnia	daleki	lekki	tak

$$\boxed{X_1} = \text{"tak"} = \{\boxed{o_1, o_2, o_3, o_7, o_9}\}$$

$$X_2 = \text{"nie"} = \{o_5, o_6, o_8\}$$

Tworzymy zbiory z kolumny „d” dla TAK i NIE

$$A: \{\underbrace{\boxed{\{o_1, o_2, o_3, o_7\}}}, \underbrace{\{o_4, o_5\}}, \underbrace{\{o_6, o_8, o_9\}}\}$$

$$\underline{X}_{1A} = \{o_1, o_2, o_3, o_7\} \qquad \underline{X}_2 A = \emptyset$$

$$\overline{X}_{1A} = \{o_1, o_2, o_3, o_7, o_6, o_8, o_9\} \qquad \overline{X}_2 A = \{o_4, o_5, o_6, o_8, o_9\}$$

$$X_1 = \overline{X}_1 - \underline{X}_1 = \{o_6, o_8, o_9\}$$

A tworzymy przy pomocy kolorków o1 itp.

Czyli kolor czerwony ma o1o2o3o7 itp.

Aproksymacja dolna jest to zbiór który mieści się CAŁY w x1 lub x2 do zbioru A

Czyli o1,o2,o3,o7 mieści się cały w x1

Aproksymacja górna oblicza się za pomocą tego ze jeśli jakieś o z zbioru A pasuje do danego zbioru w x1 to przepisuje wszystko z zbioru A

Wiec zbiór o4,o5 nie pasuje do x1 bo nie posiada o4 ani o5

Potem oblicza się różnicę aproksymacji górnej do dolnej

$$B = \{a_1, a_2\}$$

$$B: \{\{o_1, o_2, o_3, o_7\}, \{o_4, o_5, o_6, o_8, o_9\}\} \qquad \underline{X}_{2B} = \emptyset$$

$$\underline{X}_{1B} = \{o_1, o_2, o_3, o_7\}$$

$$\overline{X}_{1B} = \{o_1, o_2, o_3, o_7, o_4, o_5, o_6, o_8, o_9\} \qquad \overline{X}_{2B} = \{o_4, o_5, o_6, o_8, o_9\}$$

Tutaj widać, że nie korzystamy już z a3 więc trzeba patrzeć na wartości w wierszach. Na przykład „wysoka” i „bliski” to potem już liczysz tak samo jak wyżej.

Entropia

Example	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	Decision WillWait
x_1	Yes	No	No	Yes	Some	\$\$\$	No	Yes	French	0-10	Yes
x_2	Yes	No	No	Yes	Full	\$	No	No	Thai	30-60	No
x_3	No	Yes	No	No	Some	\$	No	No	Burger	0-10	Yes
x_4	Yes	No	Yes	Yes	Full	\$	Yes	No	Thai	10-30	Yes
x_5	Yes	No	Yes	No	Full	\$\$\$	No	Yes	French	> 60	No
x_6	No	Yes	No	Yes	Some	\$\$	Yes	Yes	Italian	0-10	Yes
x_7	No	Yes	No	No	None	\$	Yes	No	Burger	0-10	No
x_8	No	No	No	Yes	Some	\$\$	Yes	Yes	Thai	0-10	Yes
x_9	No	Yes	Yes	No	Full	\$	Yes	No	Burger	> 60	No
x_{10}	Yes	Yes	Yes	Yes	Full	\$\$\$	No	Yes	Italian	10-30	No
x_{11}	No	No	No	No	None	\$	No	No	Thai	0-10	No
x_{12}	Yes	Yes	Yes	Yes	Full	\$	No	No	Burger	30-60	Yes

dla Alt

	p	n	
→ yes	3	3	$\frac{3}{3+3}$
no	3	3	

Liczę, ile występuję yes i no pomiędzy Alt i Decision

dla Altermate :

	p	n
yes	3	3
no	3	3

- $B_{yes} = B\left(\frac{3}{6}\right) = B\left(\frac{1}{2}\right) = -\left(\frac{1}{2}\log_2\left(\frac{1}{2}\right) + \left(1-\frac{1}{2}\right)\log_2\left(1-\frac{1}{2}\right)\right) =$
  $= -\left(\left(-\frac{1}{2}\right) + \left(-\frac{1}{2}\right)\right) = 1$
- $B_{no} = B\left(\frac{3}{6}\right) = B\left(\frac{1}{2}\right) = 1$

$Gain(Alt) = 1 - Reminder(Alt) \quad \left(B\left(\frac{6}{12}\right) - Reminder(Alt)\right)$

$= 1 - \left[\frac{6}{12}\cdot 1 + \frac{6}{12}\cdot 1\right] = 1 - 1 = 0$ bit

Tworze na przykład B_yes i B_no czyli biorę wartość z yes(p) czyli 3 i potem w mianowniku jest to suma dla yes dla P i N czyli 3+3 to 6

Potem B(3/6) to B(1/2) zawsze jak się równa ½ to wychodzi 1

Ale to działa na takiej zasadzie, że ułamek co jest w nawiasie B przepisuje przed i po log a w drugim logarytmie jest to samo tylko, że dla wartości N czyli 3/6 można na to tak patrzeć tez ze jest to dopełnienie mianownika. Na przykład 1/5 to drugi log musi mieć 4/5 itp.

Końcowy bit oblicza się z wzoru 1-[(suma dla wiersza yes przez sumę wszystkich wierszy czyli 6/12)*wartość Byes+ 6/12 * wartość dla Bno] tak dostajemy wynik

Gain (Patron) = 0,541 bits I

Gain (Type) = 0 bit VI

Gain (Alt) = 0 bit VI

Gain (Bar) = 0 bit VI

Gain (Fri) = 0,022 bits IV

Gain (Hun) = 0,256 bits II

Gain (Price) = 0,187 bits III

Gain (Rain) = 0,022 bits IV

Gain (Res) = 0,022 bits IV

Gain (Est) = 0,021 bits V

Przykładowa ścieżka:

I Patron → Hun → Price → Fri → Est

II Patron → Hun → Price → Fri → Rain → Res → Est →
→ Type → Alt → Bar

Potem na końcu ustalamy najlepsza ścieżkę na przykład od największej do najmniejszej