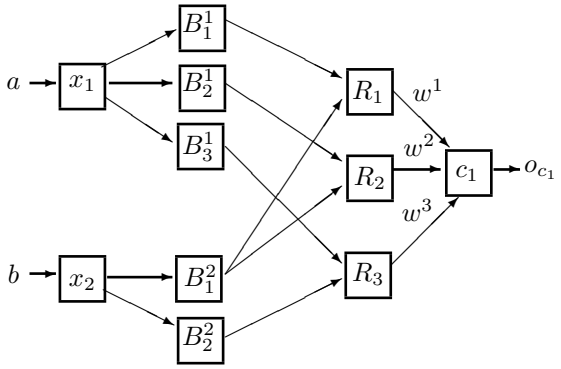
1 Hybridné neurónové siete



Definícia 1 (HNN) *Hybridná neurónová sieť je štvorvrstvová feedforward sieť* (N, W, A, O) *špecifikovaná nasledovne:*

1) $N = \bigcup_{i \in I} N_i$ *je neprázdna množina neurónov,* $I = \{1, 2, 3, 4\}$ *je indexová množina. Pre každé* i, $j \in I$ *platí:* $N_i \neq \emptyset$ *a* $N_i \cap N_j = \emptyset$ *pre* $i \neq j$.
N_1 *sa nazýva vstupná vrstva,* N_2 *fuzzifikačná vrstva,* N_3 *vrstva pravidiel (alebo inferenčná vrstva), ktorá odpovedá predpokladovým častiam pravidiel a* N_4 *výstupná vrstva, odpovedajúca dôsledkovým častiam pravidiel.*
Ďalej označíme $N_1 = \{x_1, \ldots, x_n\}$, $N_2 = \{M_j^i \mid i = 1, \ldots, n;\ j = 1, \ldots, q_i\}$, $N_3 = \{R_1, \ldots, R_K\}$, $N_4 = \{c_1, \ldots, c_m\}$. *Každý neurón* $M_j^i \in N_2$ *je spojený s nejakou hodnotou lingvistickej premennej* L_i, *číslo* q_i *predstavuje počet hodnôt tejto lingvistickej premennej.*

2) $W : N \times N \to \mathrm{R}$ *je zobrazenie definujúce štruktúru siete (prepojenia váhami) a spĺňajúce nasledujúce podmienky:*

a) Pre $n, m \in N$ *prepojenia* $W(n, m)$ *existujú iba pre* $n \in N_i$ *a* $m \in N_{i+1}$ *pre* $i = 1,\ 2,\ 3$.
Navyše $W(n, m) \in \{0, 1\}$ *pre* $n \in N_i$, $m \in N_{i+1}$, $i = 1, 2$.

b)

$$W(x_i, M_j^k) = \begin{cases} 1 & k = i \\ 0 & inak \end{cases}$$

c) Pre každé R, M_j^i, M_k^i *platí*

$$(W(M_j^i, R) = 1 \ \&\ W(M_k^i, R) = 1) \Longrightarrow k = j$$

Teda v jednom pravidle sa môže vyskytovať jediná lingvistická premenná týkajúca sa x_i.

d) Pre každé R, R' *platí*

$$(\forall\, M_j^i \quad W(M_j^i, R) = W(M_j^i, R')) \Longrightarrow R = R'$$

T.j. Predpokladové časti jednotlivých pravidiel sa odlišujú aspoň v jednej lingvistickej premennej.

3) A *definuje agregačnú funkciu* A_n *pre každý neurón* $n \in N$ *nasledovne*

a) pre $n \in N_1$ $A_n : \mathrm{R} \to \mathrm{R}$

$$a_n = A_n(ex_n) = ex_n,$$

kde ex_n *je* n*-tá zložka externého vstupu siete*

b) pre $n \in N_2$ $A_n : \mathrm{R} \to \mathrm{R}$

$$a_n = A_n(o_{n'}) = o_{n'}, \quad n' \in N_1 : W(n', n) = 1$$

c) pre $n \in N_3$ $A_n : \langle 0,1\rangle \to \langle 0,1\rangle$

$$a_n = \underset{n' \in S}{\mathbf{t}}(o_{n'}), \quad S = \{m \in N_2 : W(m,n) = 1\}$$

d) pre $n \in N_4$ $A_n : \langle 0,1\rangle \to \langle 0,1\rangle$

$$a_n = \underset{n' \in N_3}{\mathbf{s}}\{\mathbf{t}(W(n',n), o_{n'})\} \quad \textit{v prípade, že } W(n',n) \in \langle 0,1\rangle,$$

resp. $A_n : \langle 0,1\rangle \to \mathrm{R}$

$$a_n = \frac{\sum_{n' \in N_3} W(n',n) o_{n'}}{\sum_{n' \in N_3} W(n',n)}$$

4) O definuje pre každý neurón $n \in N$ *výstupnú funkciu* O_n *nasledovne:*

a) pre $n \in N_1$ $O_n : \mathrm{R} \to \mathrm{R}$

$$o_n = O_n(a_n) = a_n$$

b) pre $n \in N_2$ $O_n : \mathrm{R} \to \langle 0,1\rangle$

$$o_n = O_n(a_n) = \mu_n(a_n)$$

presnejšie: pre $M_j^i \in N_2$

$$o_{M_j^i} = O_{M_j^i}(a_{M_j^i}) = \mu_j^i(a_{M_j^i}),$$

kde μ_j^i *je funkcia príslušnosti* j*-tej hodnoty* i*-tej lingvistickej premennej* L_i

c) pre $n \in N_3$ $O_n : \langle 0,1\rangle \to \langle 0,1\rangle$

$$o_n = O_n(a_n) = a_n$$

d) pre $n \in N_4$ $O_n : \langle 0,1\rangle \to \langle 0,1\rangle$

$$o_n = O_n(a_n) = a_n,$$

resp. $O_n : \mathrm{R} \to \mathrm{R}$

$$o_n = O_n(a_n) = a_n.$$

2 Neuro–fuzzy klasifikátory

2.1 Klasifikačný problém

Jeden z problémov, na riešenie ktorého je možné využiť neurónové siete, je aj klasifikačný problém. Jeho podstata je jednoduchá.

Máme danú triedu C objektov, ktoré potrebujeme podľa ich atribútov zatriediť do m podtried $C^1, C^2, \ldots, C^m \subseteq C$. Predpokladáme pritom, že $C^1, C^2, \ldots, C^m$ tvoria rozklad triedy C a že všetky objekty $x \in C$ je možné charakterizovať n spoločnými atribútmi $a_1, a_2, \ldots, a_n \in \mathrm{R}$.

Hodnoty atribútov konkrétneho objektu x budeme označovať $\mathbf{a}^x = (a_1^x, a_2^x, \ldots, a_n^x)$. V prípade, že každú triedu C^i reprezentujeme vektorom $\mathbf{e}^i \in \mathrm{R^m}$, $\mathbf{e}^i = (\overset{1}{0}, 0, \ldots, 0, \overset{i}{1}, 0, \ldots, \overset{m}{0})$, je potom našou úlohou nájsť takú funkciu $f : \mathrm{R^n} \to \{0,1\}^m$, pre ktorú

$$f(a_1^x, a_2^x, \ldots, a_n^x) = \mathbf{e}^{k_x}, \quad k_x \in \{1, \ldots, n\}, \quad \text{pre všetky } x \in C.$$

f teda predstavuje klasifikačnú funkciu, ktorá každej n-tici atribútov $\mathbf{a}^x$ objektu x priradí m-rozmerný vektor $\mathbf{e}^{k_x}$ reprezentujúci príslušnú podtriedu C^{k_x}, do ktorej x patrí.

V prípade, že máme k dispozícii dostatočne rozsiahlu trénujúcu vzorku správne klasifikovaných príkladov, je možné využiť učiacu schopnosť neurónových sietí a používať ich ako klasifikátor.

Ak navyše existujú aspoň hrubo popísané pravidlá (zákonitosti), na základe ktorých sa zatrieďovanie vykonáva, je vhodné na aproximáciu klasifikačnej funkcie f použiť HNN. Tu si však treba uvedomiť, že HNN v skutočnosti nebude aproximovať funkciu
$f : \mathrm{R^n} \to \{0,1\}^m$, ale nejakú funkciu $f' : \mathrm{R^n} \to \langle 0,1\rangle^m$, nakoľko hybridné neurónové siete (ako sme ich definovali my) sú spojité zobrazenia. Funkciu f je potom možné vyjadriť ako $f(\mathbf{a}^x) = g(f'(\mathbf{a}^x))$, kde $g : \langle 0,1\rangle^m \to \{0,1\}^m$ je funkcia predstavujúca interpetáciu výsledku funkcie f'.

Tie HNN, ktoré sú určené na aproximáciu klasifikačných funkcií, označujeme ako neuro–fuzzy klasifikátory (skrátene NFC). V nasledujúcej definícii uvedieme odlišnosti týchto sietí od HNN z definície 2.

Definícia 2 (NFC) *Neuro–fuzzy klasifikátor je hybridná neurónová sieť, pre ktorú navyše platia nasledujúce podmienky:*

1. $W(R,c) \in \{0,1\}$, *pre všetky* $R \in N_3$ *a* $c \in N_4$
2. *pre všetky* c, c', R *platí*
$$(W(R,c) = 1 \;\&\; W(R,c') = 1) \implies c = c'$$
čiže neexistujú dve pravidlá s rovnakými predpokladovými a rôznymi uzáverovými časťami.
3. *výstupná funkcia* O_n *pre* $n \in N_4$ *je* $O_n : \langle 0,1\rangle \to \{0,1\}$
$$o_n = O_n(a_n) = g(a_n),$$
kde $g : \langle 0,1\rangle \to \{0,1\}$ *je nejaká interpretačná funkcia.*

Štruktúra NFC je, podobne ako u HNN, daná hlavne systémom lingvistických pravidiel. Tie majú v tomto prípade nasledovný tvar:

$$\mathcal{R}^j : \text{AK } a_1^x \text{ je } A_1^j \text{ A} \ldots \text{A } a_n^x \text{ je } A_n^j \text{ POTOM } \mathbf{x} \text{ patrí do } C^{l_j}.$$

V nasledujúcom sa budeme bližšie zaoberať špeciálnym prípadom, keď $m = 2$, t.j. keď je potrebné prvky zaradiť do dvoch tried.

3 Klasifikácia do dvoch tried

Je potrebné si uvedomiť, že klasifikačnú funkciu

$$f : \mathrm{R^n} \to \{0,1\}^2, f(\mathbf{a}^x) = (c_1,\, c_2)$$

je možné nahradiť funkciou $\varphi : \mathrm{R^n} \to \{0,1\}$ takou, že $\varphi(\mathbf{a}^x) = c_1$ (resp. $\varphi(\mathbf{a}^x) = c_2$). Výsledok $c_1 = 1$ pritom znamená, že $x \in C^1$ a $c_1 = 0$, že $x \in C^2$ (v druhom prípade analogicky pre c_2).

Uvažujme funkciu $\varphi : X \to \{0,1\}$, $X \subseteq \mathrm{R^n}$, $\varphi(\mathbf{a}^x) = c_1$ takú, že

$$\exists\, K < \infty\, :\, \{\mathbf{a}^x \in X\, :\, \varphi(\mathbf{a}^x) = 1\} = \bigcup_{j=1}^{K} (\mathbf{k}^j,\, \mathbf{l}^j), \quad \mathbf{k}^j,\, \mathbf{l}^j \in X.$$

Označme Φ triedu takýchto funkcií. V nasledujúcom tvrdení ukážeme, že trieda funkcií reprezentovaných neuro–fuzzy klasifikátormi obsahuje triedu Φ. Zároveň dáme popis, ako k ľubovoľnej funkcii $\varphi \in \Phi$ zostrojiť NFC, ktorý predstavuje zobrazenie φ.

Označme teda $\mathcal{N}$ triedu NFC z definície (2), pre ktoré platí:

1. $|N_1| = n,\ |N_2| = nK,\ |N_3| = K,\ |N_4| = 1$
2. agregačná funkcia je

$$\begin{aligned} \text{pre } n \in N_3 \quad a_n &= \min_{n' \in S} (o_{n'}), \quad S = \{m \in N_2 \mid W(n', n) = 1\} \\ \text{pre } n \in N_4 \quad a_n &= \max_{n' \in N_3} \{\min(W(n', n), o_{n'})\} \end{aligned}$$

3. funkcie príslušnosti sú trojuholníkového tvaru

$$\mu(z) = \begin{cases} \frac{z-u}{v-u} & z \in \langle u, v\rangle \\ \frac{w-z}{w-v} & z \in (v, w\rangle \\ 0 \quad \text{inak} \end{cases}$$

4. výstupná funkcia pre $n \in N_4$ (interpretačná funkcia) je

$$o_n = O_n(a_n) = g(a_n) = \begin{cases} 1 & a_n > 0 \\ 0 & a_n = 0. \end{cases}$$

Veta 1 $\Phi = \mathcal{N}$.

Dôkaz: i) $\Phi \subseteq \mathcal{N}$
Nech φ je ľubovoľná funkcia z Φ. Potom

$$\exists\, K < \infty\, :\, \{\mathbf{a}^x \in X\, :\, \varphi(\mathbf{a}^x) = 1\} = \bigcup_{j=1}^{K} (\mathbf{k}^j,\, \mathbf{l}^j), \quad \mathbf{k}^j,\, \mathbf{l}^j \in X,\ X \subseteq \mathrm{R^n}.$$

Pre každý interval $(\mathbf{k}^j,\, \mathbf{l}^j)$ zostrojíme pravidlo $\mathcal{R}^j$ nasledovne:

$$\mathcal{R}^j\ :\ \text{AK}\ a_1^x\ \text{je}\ M_1^j\ \text{A} \ldots \text{A}\ a_n^x\ \text{je}\ M_n^j\ \text{POTOM}\ x\ \text{patrí do}\ C^1.$$

$M_i^j \in \mathcal{M}_i\ j = 1 \ldots, K$, pričom $\mathcal{M}_i,\ i = 1, \ldots, n$ je systém fuzzy množín, ktoré predstavujú nejaké hodnoty lingvistickej premennej L_i spojenej s i-tym atribútom príkladu x. Tieto fuzzy množiny budú trojuholníkového tvaru so stredom v strede intervalu $(k_i^j,\, l_i^j)$ a okrajmi v bodoch k_i^j a l_i^j:

$$M_i^j(z) = \begin{cases} \frac{z-u_i^j}{v_i^j-u_i^j} & z \in \langle u_i^j,\, v_i^j\rangle \\ \frac{w_i^j-z}{w_i^j-v_i^j} & z \in (v_i^j,\, w_i^j\rangle \\ 0 \quad \text{inak}, \end{cases}$$

kde $u_i^j = k_i^j,\ v_i^j = \frac{k_i^j + l_i^j}{2},\ w_i^j = l_i^j$.

1) Nech $\mathbf{a}^x \in X$ je také, že $\varphi(\mathbf{a}^x) = 1$. Potom

$$\begin{aligned} & \exists\, j \in \{1, \ldots, K\} : \mathbf{a}^x \in (\mathbf{k}^j, \mathbf{l}^j) \\ \Longrightarrow\ & \exists\, j \in \{1, \ldots, K\}\, \forall\, i \in \{1, \ldots, n\} : a_i^x \in (k_i^j, l_i^j) \\ \Longrightarrow\ & \exists\, j \in \{1, \ldots, K\}\, \forall\, i \in \{1, \ldots, n\} : M_i^j(a_i^x) > 0. \end{aligned}$$

Výstup z tohoto j-teho pravidla bude

$$o_{R_j} = \min_{i=1,\ldots,n} M_i^j(a_i^x) > 0.$$

Výstup siete bude preto $o_{c_1} = 1$.

2) Nech teraz $\mathbf{a}^x \in X$ je také, že $\varphi(\mathbf{a}^x) = 0$. Potom

$$\begin{aligned} & \forall\, j \in \{1, \ldots, K\}\, \exists\, i \in \{1, \ldots, n\} : a_i^x \notin (k_i^j, l_i^j) \\ \Longrightarrow\ & \forall\, j \in \{1, \ldots, K\}\, \exists\, i \in \{1, \ldots, n\} : M_i^j(a_i^x) = 0. \end{aligned}$$

Výstup každého pravidla $\mathcal{R}^j$, pre $j \in \{1, \ldots, K\}$ bude

$$o_{R_j} = \min_{i=1,\ldots,n} M_i^j(a_i^x) = 0.$$

Preto výstupom siete bude $o_{c_1} = 0$.

ii) $\Phi \supseteq \mathcal{N}$

Nech sieť z $\mathcal{N}$ je ľubovoľná. Každý neurón M_i^j v druhej vrstve zodpovedá fuzzy množine trojuholníkového tvaru s parametrami $u_i^j,\ v_i^j,\ w_i^j$. Z týchto neurónov zostrojíme K n-rozmerných intervalov $(\mathbf{k}^j, \mathbf{l}^j)$ nasledovne:

$$k_i^j = u_i^j, \quad l_i^j = w_i^j, \qquad i = 1, \ldots, n,$$

a vezmeme $\varphi \in \Phi$ takú, že

$$\{\mathbf{a}^x \in X : \varphi(\mathbf{a}^x) = 1\} = \bigcup_{j=1}^{K} (\mathbf{k}^j, \mathbf{l}^j).$$

1) Nech $\mathbf{a}^x$ je také, že výstup siete na vstupe $\mathbf{a}^x$ je $o_{c_1} = 1$. Potom

$$\begin{aligned} & \exists\, j \in \{1, \ldots, K\} : o_{R_j} > 0 \\ \Longrightarrow\ & \exists\, j \in \{1, \ldots, K\}\, \forall\, i \in \{1, \ldots, n\} : M_i^j(a_i^x) > 0 \\ \Longrightarrow\ & \exists\, j \in \{1, \ldots, K\}\, \forall\, i \in \{1, \ldots, n\} : a_i^x \in (k_i^j, l_i^j) \\ \Longrightarrow\ & \mathbf{a}^x \in (\mathbf{k}^j, \mathbf{l}^j) \Longrightarrow \varphi(\mathbf{a}^x) = 1. \end{aligned}$$

2) Nech teraz $\mathbf{a}^x$ je také, že výstup siete na vstupe $\mathbf{a}^x$ je $o_{c_1} = 0$. Potom

$$\begin{aligned} & \forall\, j \in \{1, \ldots, K\} : o_{R_j} = 0 \\ \Longrightarrow\ & \forall\, j \in \{1, \ldots, K\}\, \exists\, i \in \{1, \ldots, n\} : M_i^j(a_i^x) = 0 \\ \Longrightarrow\ & \forall\, j \in \{1, \ldots, K\}\, \exists\, i \in \{1, \ldots, n\} : a_i^x \notin (k_i^j, l_i^j) \\ \Longrightarrow\ & \forall\, j \in \{1, \ldots, K\}\, \mathbf{a}^x \notin (\mathbf{k}^j, \mathbf{l}^j) \Longrightarrow \varphi(\mathbf{a}^x) = 0 \end{aligned}$$

Teda $\Phi = \mathcal{N}$.
čo sme potrebovali ukázať.

Toto tvrdenie dáva návod ako zostrojiť NFC v prípade, že klasifikačná funkcia φ je z Φ. Je však užitočné aj v niektorých prípadoch, keď $\varphi \notin \Phi$. Stačí si uvedomiť, že pri vytváraní klasifikátora v podobe

neurónovej siete máme vždy k dispozícii iba nejakú konečnú trénujúcu vzorku. Tá pozostáva z vybraných príkladov a ich správnych klasifikácií a predstavuje vlastne nejakú diskrétnu klasifikačnú funkciu φ_D. Tú už môžeme považovať za funkciu z Φ a teda zostrojiť k nej NFC tak, že bude schopný správne klasifikovať všetky príklady z trénujúcej vzorky.

Praktické použitie tohoto tvrdenia je však obmedzené. Problémom totiž je určenie intervalov $(\mathbf{k}^j, \mathbf{l}^j)$. Ak na ich získanie využijeme fakt, že trénujúca vzorka je konečná a teda vždy môžeme množinu pokryť konečným zjednotením intervalov $(\mathbf{k}^j, \mathbf{l}^j)$, dosiahneme síce 100% -tnú úspešnosť pri klasifikácii trénujúcich príkladov, pri klasifikácii iných príkladov však môže dochádzať k častým chybám. Bude to spôsobené tým, že sieť nevznikla *učením sa*, ale našou konštrukciou, ktorá sa takýmto výberom intervalov $(\mathbf{k}^j, \mathbf{l}^j)$ príliš sústredila na detaily v štruktúre trénujúcej vzorky. Sieť týmto stratila svoju schopnosť zovšeobecňovať. Iný, vhodnejší, postup pri výbere intervalov $(\mathbf{k}^j, \mathbf{l}^j)$ by mohol byť nasledovný: Analýzou trénujúcej vzorky najprv zistiť *zhluky príkladov*, v ktorých väčšina patrí do jednej triedy. Na základe týchto *zhlukov* potom určiť intervaly $(\mathbf{k}^j, \mathbf{l}^j)$ a tým aj počiatočné nastavenie siete. Problémom pri tomto spôsobe však ostáva nájdenie vhodných *zhlukov*, pri ktorých by sieť získala relatívne dobré počiatočné nastavenie a zachovala by si aj istú schopnosť zovšeobecňovania.

V nasledujúcej stati uvedieme niekoľko algoritmov, ktoré je možné použiť na dotvorenie (prípadne úplné vytvorenie) štruktúry NFC z trénujúcej vzorky.

4 Metódy vytvárania pravidiel z trénujúcich dát

Pri vytváraní NFC je nutné do jeho štruktúry zabudovať už známe informácie, vyjadrujúce štruktúru trénujúcej vzorky, vo forme AK-POTOM pravidiel.

V mnohých prípadoch však takéto informácie k dispozícii nie sú, prípadne sú neúplné, t.j. nezachytávajú všetky potrebné zákonitosti. Klasifikátor musí v týchto prípadoch vytvoriť vlastný, prípadne doplniť existujúci súbor pravidiel tak, aby bol schopný naučiť sa správne rozpoznávať trénujúce vzorky.

Ukážeme niekoľko algoritmov, ktoré je možné použiť na extrahovanie pravidiel z trénujúcich dát. Algoritmy sa odlišujú jednak v tom, či pokrytie priestorov parametrov fuzzy množinami nechávajú nemenné, alebo ho modifikujú, ako aj v tom, ako sa dokážu vysporiadať s nejednoznačnosťou pravidiel.

4.1 Princíp vytvárania pravidiel, nejednoznačnosť pravidla

Predpokladajme, že máme danú trénujúcu vzorku T príkladov $(\mathbf{x}, \mathbf{c})$ t.j. $\mathbf{x} = (x_1, \ldots, x_n)$, $x_i \in X_i \subset \mathrm{R}$, $i = 1, \ldots, n$. Predpokladajme tiež, že pre každý parameter máme definovaný neprázdny systém $\mathcal{M}_i$ fuzzy množín definovaných na X_i. Potrebujeme vytvoriť $\mathcal{S}$ systém pravidiel $\mathcal{R}^j$ typu

$$\mathcal{R}^j : \text{AK } x_1 \text{ je } A_1^j \text{ A } \ldots \text{ A } x_n \text{ je } A_n^j \text{ POTOM } \mathbf{x} \in C^{l_j}, \quad A_i^j \in \mathcal{M}_i \ i = 1, \ldots, n, \tag{1}$$

pomocou ktorého je možné každý príklad trénujúcej vzorky zaradiť do správnej triedy.

Princíp ako k danému príkladu $(\mathbf{x}, \mathbf{c})$, $\mathbf{c} = \mathbf{e}^k$, vytvoríme pravidlo $\mathcal{R}$ je nasledovný:

Za A_i vezmeme takú fuzzy množinu N_i zo systému $\mathcal{M}_i$, že stupeň príslušnosti $\mu_{N_i}(x_i)$ prvku x_i do N_i je maximálny spomedzi všetkých stupňov príslušnosti prvku x_i do fuzzy množín v $\mathcal{M}_i$.

Za triedu C vezmeme samozrejme tú, ktorá odpovedá vektoru $\mathbf{c} = \mathbf{e}^k$, t.j. C^k. Pravidlo $\mathcal{R}$ teda bude vyzerať nasledovne:

$$\mathcal{R} : \text{AK } x_1 \text{ je } N_1 \text{ A } \ldots \text{ A } x_n \text{ je } N_n \quad \text{POTOM} \quad \mathbf{x} \in C^k, \tag{2}$$

kde

$$\mu_{N_i}(x_i) = \max_{M_i \in \mathcal{M}_i} \mu_{M_i}(x_i).$$

To, že za A_i vyberáme množinu s maximálnym stupňom príslušnosti x_i do nej, zodpovedá snahe vybrať najsilnejšiu závislosť spomedzi všetkých, t.j. vytvoriť pravidlo, ktoré je zo všetkých možných pravidiel pre daný príklad $(\mathbf{x}, \mathbf{c})$ najsilnejšie. Do pravidla teda vyberáme najsilnejšiu možnú kombináciu fuzzy množín.

Problém, ktorý pri vytváraní pravidiel vzniká, je ten, že v trénujúcej vzorke môžu existovať príklady patriace do rôznych tried, pre ktoré by sme mali vybrať rovnakú kombináciu fuzzy množín. Takto by sme získali protirečivé pravidlá, ktoré sa síce v predpokladovej časti rovnajú, ale v uzáverovej časti sa líšia. Takéto pravidlá nazveme **nejednoznačné** alebo sporné. Výskyt týchto pravidiel závisí jednak od štruktúry systémov $\mathcal{M}_i$, konkrétne od počtu fuzzy množín v $\mathcal{M}_i$ a od definícií ich funkcií príslušnosti

a taktiež od trénujúcej vzorky T. Ak je T sporná (teda existuje $(\mathbf{x}^1, \mathbf{c}^1) \in$ T a $(\mathbf{x}^2, \mathbf{c}^2) \in$ T také, že $\mathbf{x}^1 = \mathbf{x}^2$ & $\mathbf{c}^1 \neq \mathbf{c}^2$), nie je možné vyhnúť sa vzniku nejednoznačného pravidla.

Algoritmy použiteľné na získavanie pravidiel z trénujúcich dát, ktoré uvedieme v tejto časti, sa odlišujú práve spôsobom, akým sa vysporiadavajú s nejednoznačnými pravidlami. Možnosti sú v zásade štyri:

1. Nejednoznačné pravidlo nezaradíme do súboru pravidiel. Tento prístup je nevhodný, nakoľko výsledky klasifikátora pri príkladoch, ktoré viedli k vytvoreniu tohoto nejednoznačného pravidla, je ťažko určiť a navyše nemusia zodpovedať žiadnej z tried, ktoré figurovali ako možné uzáverové časti tohoto pravidla.

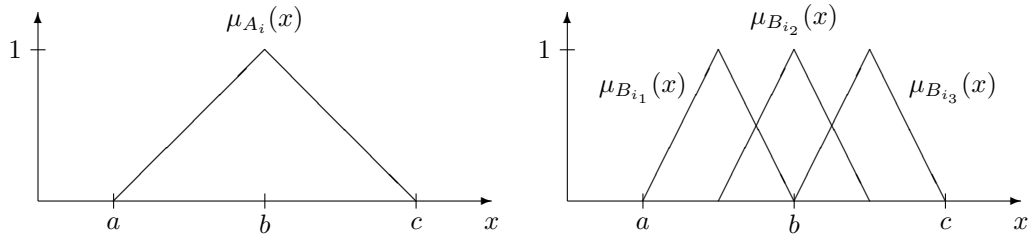
2. Zavedieme novú triedu C_{NC} - triedu neklasifikovateľných príkladov a uzáverové časti všetkých nejednoznačných pravidiel zmeníme na C_{NC}. Toto riešenie je v našom prípade tiež nevhodné, pretože zbavuje neurónovú sieť svojej funkcie. Sieť totiž nedostane šancu naučiť sa klasifikovať tieto sporné príklady.

3. Pre každé nejednoznačné pravidlo sa na základe nejakého kritéria rozhodneme, ktorú triedu vyberieme do uzáverovej časti pravidla. Kritériom môže byť napríklad počet výskytov jednotlivých tried v uzáverovej časti pravidla, súčet síl pravidiel vytvorených pre príklady patriace do tej-ktorej triedy, prípadne niečo iné. Takéto riešenie sa používa napríkad v [6].

4. Každú fuzzy množinu A_i vyskytujúcu sa v predpokladovej časti pravidla nahradíme niekoľkými fuzzy množinami B_{i_j}, $j = 1, \ldots, b_i$ $b_i \geq 2$ tak, aby

$$\bigcup_{j=1,\ldots,b_i} Supp\, B_{i_j} = Supp\, A_i \tag{3}$$

a opakujeme proces extrahovania pravidiel. Príklad nahradenia množiny A_i troma fuzzy množinami B_{i_j}, $j = 1, 2, 3$ je uvedený na obr. 1.



Obr. 1: Príklad nahradenia fuzzy množiny A_i tromi fuzzy množinami B_{i_j}, pre $j = 1, 2, 3$.

Tento spôsob pri dostatočne veľkom počte opakovaní a za predpokladu, že trénujúca vzorka nie je sporná, úplne odstráni nejednoznačné pravidlá. Je však dobré si uvedomiť, že v našom prípade nie je cieľom za každú cenu vytvoriť súbor jednoznačných pravidiel. Takýto súbor totiž vystihuje štruktúru trénujúcej vzorky až príliš detailne a dáva len malú možnosť využiť tieto pravidlá na klasifikáciu príkladov nezahrnutých do trénujúcej vzorky. Pri vytváraní súboru pravidiel pre NFC je cieľom vytvoriť také pravidlá, ktoré umožňujú úspešnú klasifikáciu príkladov nezahrnutých do trénujúcej vzorky aj za cenu nižšej úspešnosti pri klasifikácii trénujúcich príkladov. Z tohoto dôvodu je pri tomto spôsobe vhodné zaviesť nejakú podmienku, na základe ktorej je možné rozhodnúť, či tú-ktorú fuzzy množinu nahradiť novými množinami, alebo nie. Do úvahy je pritom možné brať obe kritériá uvádzané v prípade 3), ako aj maximálny počet, koľkokrát je povolený proces nahrádzania fuzzy množín novými a následného extrahovania pravidiel.

4.2 Algoritmy vytvárania bázy znalostí z trénujúcich dát

Vo všetkých algoritmoch, ktoré v tejto časti uvedieme, uvažujeme symbolickú konštantu MR, ktorá určuje maximálny počet pravidiel, ktoré dovolíme vyrobiť. Symbolom $ANT(\mathcal{R})$ budeme označovať n-ticu fuzzy množín z predpokladovej časti pravidla $\mathcal{R}$.

4.2.1 Algoritmus 1.

Algoritmus, ktorý predstavíme ako prvý, je v podstate nepoužiteľný. Uvádzame ho, lebo predstavuje najjednoduchšie riešenie problému extrahovania pravidiel z trénujúcich dát.

Algoritmus 1:

K 1: Vezmi ďalší príklad z trénujúcej vzorky.

K 2: Vytvor k nemu pravidlo $\mathcal{R}$ podľa vzťahu (2).

K 3: Ak v súbore pravidiel $\mathcal{S}$ neexistuje pravidlo s predpokladovou časťou $ANT(\mathcal{R})$, zaraď $\mathcal{R}$ do $\mathcal{S}$.

K 4: Ak boli spracované všetky príklady z T alebo $|\mathcal{S}| = \mathrm{MR}$, koniec. Inak opakuj celý proces prechodom na krok 1.

Tento algoritmus je uvádzaný v [6] pod názvom *Simple*. Názov je výstižný, nakoľko nejednoznačnosť pravidla je v tomto prípade jednoducho riešená tak, že za uzáverovú časť sporného pravidla sa vezme tá, ktorá sa vytvorí ako prvá. Úspech *učenia sa* NFC v tomto prípade silne závisí na poradí, v akom sa príklady nachádzajú v trénujúcej vzorke.

4.2.2 Algoritmus 2.

Algoritmus, ktorého princíp je čerpaný z [6], odstraňuje hlavný nedostatok predchádzajúceho algoritmu použitím kritéria, ktoré zohľadňuje silu pravidiel vytvorených pre jednotlivé príklady, ako aj početnosť ich použitia. Význam symbolov použitých v algoritme je nasledujúci:

- $(\mathbf{x}^k, \mathbf{c}^k) \ldots k$-ty príklad z trénujúcej vzorky T, $\mathbf{x}^k \in \mathrm{R}^n$ a $\mathbf{c}^k \in \{0,1\}^m$
- $\mathcal{S} \ldots$ systém pravidiel, ktorý chceme vytvoriť, resp. doplniť (na začiatku algoritmu už môže obsahovať niekoľko, predpokladáme menej ako MR, pravidiel)
- $\mathcal{S}' \ldots$ pomocný súbor pravidiel
- $\mathcal{R}^i \ldots$ AK-POTOM pravidlo typu (1)
- $FL^i(\mathbf{x}^k) \ldots$ sila pravidla $\mathcal{R}^i$ pre $(\mathbf{x}^k, \mathbf{c}^k)$
- P^i, $S^i \ldots$ polia veľkosti m definované pre pravidlo $\mathcal{R}^i$.

Algoritmus 2:

K 0: $k = 1$, $\mathcal{S}' = \emptyset$.

K 1: Vyber k-ty príklad $(\mathbf{x}^k, \mathbf{c}^k)$ z trénujúcej vzorky T.

K 2: Podľa (2) vyrob ku $(\mathbf{x}^k, \mathbf{c}^k)$ pravidlo $\mathcal{R}^t$ a urči jeho silu $FL^t(\mathbf{x}^k)$.

K 3: Ak v $\mathcal{S}'$ existuje pravidlo $\mathcal{R}^i$ také, že $ANT(\mathcal{R}^i) = ANT(\mathcal{R}^t)$, tak
$P^i[j] = P^i[j] + 1$,
$S^i[j] = S^i[j] + FL^t(\mathbf{x}^k)$,
pričom $j \in \{1, \ldots, m\}$ je také, že $\mathbf{c}^k = \mathbf{e}^j$.

Inak zaraď $\mathcal{R}^t$ do $\mathcal{S}'$ a inicializuj polia P^t, S^t nasledovne:

$$P^t[l] = \begin{cases} 1 & l = j \\ 0 & l \neq j, \end{cases}$$

$$S^t[l] = \begin{cases} FL^t(\mathbf{x}^k) & l = j \\ 0 & l \neq j, \end{cases}$$

pričom $j \in \{1, \ldots, m\}$ také, že $\mathbf{c}^k = \mathbf{e}^j$.

K 4: Ak boli spracované všetky príklady z T, pokračuj krokom 5.
Inak $k = k + 1$ a prejdi opäť na krok 1

K 5: Pre každé pravidlo $\mathcal{R}^i \in \mathcal{S}'$ nastav jeho uzáverovú časť na C^v, pričom $v \in \{1, \ldots, m\}$ je také, že $S^i[v] = \max_{j=1,\ldots,m} S^i[j]$.

K 6: Ak $|\mathcal{S}| + |\mathcal{S}'| > \mathrm{MR}$, tak pre každé $\mathcal{R}^i \in \mathcal{S}'$ urči $p^i = P^i[v]$, $v \in \{1, \ldots, m\}$ také,
že $S^i[v] = \max_{j=1,\ldots,m} S^i[j]$ a usporiadaj pravidlá v $\mathcal{S}'$ zostupne
podľa p^i a do $\mathcal{S}$ pridaj prvých $\mathrm{MR} - |\mathcal{S}|$ pravidiel z $\mathcal{S}'$.
Inak do $\mathcal{S}$ pridaj všetky pravidlá z $\mathcal{S}'$.

V stručnosti popíšeme činnosť algoritmu:
V kroku 0, až kroku 4 prebieha analýza trénujúcej vzorky, počas ktorej sa postupne vytvára pomocný súbor pravidiel $\mathcal{S}'$. Pri vstupe do kroku 5 potom platia nasledujúce štyri podmienky:

1. Systém $\mathcal{S}'$ obsahuje pravidlá s navzájom rôznymi predpokladovými časťami.
2. $\mathcal{S}'$ je maximálny v tom zmysle, že $\forall\,(\mathbf{x}, \mathbf{c}) \in \mathrm{T} \quad \exists\,\mathcal{R}^j \in \mathcal{S}' \;:\; ANT(\mathcal{R}^j) = ANT(\mathcal{R})$,
kde $\mathcal{R}$ je pravidlo vytvorené k $(\mathbf{x}, \mathbf{c})$ podľa (2).
3. Pre každé $\mathcal{R}^i \in \mathcal{S}'$, $P^i[j]$ obsahuje počet príkladov $(\mathbf{x}, \mathbf{c}) \in \mathrm{T}$, ktoré viedli k vytvoreniu pravidla $\mathcal{R}$ s rovnakou predpokladovou časťou ako má $\mathcal{R}^i$ a ktoré patrili do triedy C^j.
4. Pre každé $\mathcal{R}^i \in \mathcal{S}'$, $S^i[j]$ obsahuje súčet síl pravidiel vytvorených k tým príkladom, ktoré patrili do triedy C^j a viedli k vytvoreniu pravidla s rovnakou predpokladovou časťou, ako má $\mathcal{R}^i$.

V kroku 5 sa vhodne nastavia uzáverové časti pravidiel zo systému $\mathcal{S}'$. Nakoniec sa v kroku 6 súbor pravidiel $\mathcal{S}$ doplní o potrebný (resp. povolený) počet pravidiel zo súboru $\mathcal{S}'$.

4.2.3 Algoritmus 3.

Predchádzajúce dva algoritmy riešia problém vytvárania, resp. doplňania súboru pravidiel $\mathcal{S}$ v prípadoch, keď sú známe systémy $\mathcal{M}_i$, $\quad i = 1, \ldots, n$, teda v prípadoch, keď sú vopred známe informácie o počte a približnom tvare fuzzy množín v jednotlivých systémoch $\mathcal{M}_i$.

Môže sa však stať, že potrebujeme vytvoriť klasifikátor a trénujúca vzorka predstavuje jediný zdroj informácií, ktorý máme k dispozícii. V takomto prípade je potrebné z trénujúcich dát vyextrahovať ako štruktúru jednotlivých systémov $\mathcal{M}_i$, tak aj pravidlá, na základe ktorých by sme chceli príklady zatrieďovať.

Algoritmus 3 rieši tento problém postupným *vnáraním sa do štruktúry trénujúcej vzorky.* Princíp algoritmu je nasledovný: Spočiatku sa priestory parametrov rovnomerne pokryjú niekoľkými (nie mnohými) fuzzy množinami a začne sa vytvárať súbor pravidiel. Ak vytvorený systém pravidiel nebude vyhovujúci vzhľadom na počet a štruktúru nejednoznačných pravidiel, ktoré obsahuje, každá fuzzy množina, vyskytujúca sa v predpokladovej časti niektorého nejednoznačného pravidla, ktoré svojou štruktúrou nevyhovuje, sa nahradí niekoľkými novými fuzzy množinami podľa (3) a vytvorí sa nový súbor pravidiel. Tento proces nahradzovania fuzzy množín novými a následného znovuvytvárania pravidiel sa opakuje, až kým súbor pravidiel nie je vyhovujúci, resp. pokiaľ nie je prekročená povolená *hĺbka vnorenia.* Potom sa na základe rovnakého kritéria ako v predchádzajúcom algoritme vyberie nie viac ako MR najlepších pravidiel.

V tomto algoritme označuje symbol:

- MD ...
- d ... *hĺbku vnorenia*, t.j. počet, koľkokrát prebehol proces nahradzovania fuzzy množín novými MD ... maximálnu hĺbku vnorenia
- m_i ... počet fuzzy množín, ktorými sa spočiatku pokryje priestor X_i
- A_i ... fuzzy množinu zo systému $\mathcal{M}_i$
- $\mathcal{P}$ ... podmienku vyjadrujúcu vhodnosť systému pravidiel
- b ... počet fuzzy množín, koľkými budeme nahrádzať jednotlivé fuzzy množiny zo systémov $\mathcal{M}_i$, $\quad i = 1, \ldots, n$.

Význam ostatných symbolov je rovnaký ako v predchádzajúcom algoritme. O dvoch možných podobách podmienky $\mathcal{P}$ sa zmienime po uvedení algoritmu.

Algoritmus 3:

K 0: Pre každé $i = 1, \ldots, n$ zostroj systém $\mathcal{M}_i$, obsahujúci m_i fuzzy množín $A_{i_1}, \ldots, A_{i_{m_i}}$, ktoré rovnomerne pokrývajú priestor X_i.

K 1: $k = 1$, $d = 0$, $\mathcal{S}' = \emptyset$, $\mathcal{S} = \emptyset$.

K 2: Vyber k-ty príklad $(\mathbf{x}^k, \mathbf{c}^k)$ z trénujúcej vzorky T.

K 3: Podľa (2) vyrob ku $(\mathbf{x}^k, \mathbf{c}^k)$ pravidlo $\mathcal{R}^t$ a urči jeho silu $FL^t(\mathbf{x}^k)$.

K 4: Ak v $\mathcal{S}'$ existuje pravidlo $\mathcal{R}^i$ také, že $ANT(\mathcal{R}^i) = ANT(\mathcal{R}^t)$,
tak
$P^i[j] = P^i[j] + 1$

$S^i[j] = S^i[j] + FL^t(\mathbf{x}^k)$,

pričom $j \in \{1, \ldots, m\}$ je také, že $\mathbf{c}^k = \mathbf{e}^j$.
Inak zaraď $\mathcal{R}^t$ do $\mathcal{S}'$ a inicializuj polia P^t, S^t nasledovne:

$$P^t[l] = \begin{cases} 1 & l = j \\ 0 & l \neq j, \end{cases}$$

$$S^t[l] = \begin{cases} FL^t(\mathbf{x}^k) & l = j \\ 0 & l \neq j, \end{cases}$$

pričom $j \in \{1, \ldots, m\}$ také, že $\mathbf{c}^k = \mathbf{e}^j$.

K 5: Ak boli spracované všetky príklady z T, pokračuj krokom 6.
Inak k = k+1 a prejdi opäť na krok 2.

K 6: Ak $\mathcal{S}'$ nevyhovuje podmienke $\mathcal{P}$ a ak $d <$ MD,
tak d = d+ 1, zostroj množinu indexov I tých pravidiel z $\mathcal{S}'$, ktoré nevyhovujú podmienke $\mathcal{P}$.
Každú fuzzy množinu $A \in \mathcal{M}_i$, $i = 1, \ldots, n$, ktorá sa nachádza v predpokladovej časti niektorého nejednoznačného pravidla $\mathcal{R}^l$, $l \in I$ nahraď novými fuzzy množinami $B_1, \ldots, B_b$, $b \geq 2$
podľa (3) a opakuj celý proces prechodom na krok 1.
Inak pokračuj krokom 7.

K 7: Pre každé pravidlo $\mathcal{R}^i \in \mathcal{S}'$ nastav jeho uzáverovú časť na C^v, pričom $v \in \{1, \ldots, m\}$ je také, že $S^i[v] = \max_{j=1,\ldots,m} S^i[j]$.

K 8: Ak $|\mathcal{S}'| >$ MR, tak pre každé $\mathcal{R}^i \in \mathcal{S}'$ urči $p^i = P^i[v]$, pričom $v \in \{1, \ldots, m\}$ je také, že $S^i[v] = \max_{j=1,\ldots,m} S^i[j]$ a usporiadaj pravidlá v $\mathcal{S}'$ zostupne podľa p^i
a do $\mathcal{S}$ zaraď prvých MR pravidiel z $\mathcal{S}'$.
Inak do $\mathcal{S}$ zaraď všetky pravidlá z $\mathcal{S}'$.

Úspešnosť tohoto algoritmu závisí od vhodnej voľby parametrov MD, m_i (v kroku 0), b (v kroku 6), ktoré ovplyvňujú štruktúru výsledného pokrytia priestorov X_i fuzzy množinami z $\mathcal{M}_i$, ako aj od podmienky $\mathcal{P}$, ktorá určuje vhodnosť súboru pravidiel $\mathcal{S}'$ (v kroku 6). Z dôvodu lepšej interpretácie fuzzy množín výsledného pokrytia je vhodné voliť hodnoty parametrov m_i a b z množiny $\{3, 5, 7\}$. (Tri fuzzy množiny môžu predstavovať hodnoty Malý, Stredný, Veľký. Analogické hodnoty je možné prisúdiť piatim, resp siedmim množinám.) Podmienka $\mathcal{P}$ vhodnosti súboru pravidiel $\mathcal{S}'$ môže vyzerať nasledovne:
$\mathcal{S}'$ je vyhovujúci, ak pre každé pravidlo $\mathcal{R}^i \in \mathcal{S}'$ platí aspoň jedna z dvoch nerovností:

$$\exists\, v \in \{1, \ldots, m\} \;\; \forall\, j \in \{1, \ldots, m\},\, j \neq v,\, S^i[j] \neq 0 \,:\, \frac{S^i[v]}{S^i[j]} \geq \alpha \tag{4}$$

$$\sum_{j=1}^{m} P^i[j] \leq \beta. \tag{5}$$

Čiže za vyhovujúci súbor prehlásime taký, v ktorom pre každé sporné pravidlo platí, že sa uplatní v najviac β príkladoch trénujúcej vzorky alebo pre neho existuje jedna *dominantná* trieda spomedzi tých, ktoré prichádzajú do úvahy pre jeho uzáverovú časť.

Niekedy je výhodné za vyhovujúci súbor pravidiel $\mathcal{S}'$ považovať ten, ktorý spĺňa nasledujúcu podmienku:

$$\sum_{i \in N} \sum_{j=1}^{m} P^i[j] \leq \gamma,$$

kde N je množina indexov nejednoznačných pravidiel. Tá totiž hovorí, že klasifikátor nesprávne rozpozná nanajvýš γ príkladov trénujúcej vzorky.

Použitie tohoto algoritmu si ukážeme na príklade. Predpokladajme, že potrebujeme zostrojiť NFC, ktorý by zatrieďoval vstupné príklady do m tried $C^1, \ldots, C^m$. Predpokladajme tiež, že jediný zdroj informácií, ktorý máme k dispozícii, je trénujúca vzorka T príkladov $(\mathbf{x}, \mathbf{c})$ s n parametrami. Skôr než môžeme spustiť učiaci algoritmus, potrebujeme určiť jednak štruktúru klasifikátora a jednak počiatočné nastavenie váh a parametrov výstupných funkcií neurónov z druhej vrstvy.

Keďže príklady z T pozostávajú z n parametrov a potrebujeme ich zatrieďovať do m tried, bude počet neurónov vo vstupnej vrstve n a vo výstupnej vrstve m. ďalšie údaje sú už výsledkom algoritmu 3. Neuróny inferenčnej vrstvy označíme R^k. Ich počet bude $|\mathcal{S}|$. Neuróny vo fuzzifikačnej vrstve budú odpovedať fuzzy množinám zo systémov $\mathcal{M}_i$, $i = 1, \ldots, n$, ich počet je teda $\sum_{i=1}^{n} q_i$, $q_i = |\mathcal{M}_i|$. Neurón odpovedajúci fuzzy množine $A_{i_j} \in \mathcal{M}_i$ označíme M_j^i. Prepojenia váhami medzi prvou a druhou a druhou a treťou vrstvou sú určené nasledovne:

$$W(x^k, M_j^i) = \begin{cases} 1 & k = i \\ 0 & k \neq i, \end{cases}$$

$$W(M_j^i, R^k) = \begin{cases} 1 & \text{ak } A_{i_j} \text{ sa nachádza v predpokladovej časti pravidla } \mathcal{R}^k \\ 0 & \text{inak.} \end{cases}$$

Posledné, čo ešte potrebujeme určiť, je počiatočné nastavenie parametrov výstupných funkcií neurónov fuzzifikačnej vrstvy. Tie získame tak, že za výstupnú funkciu neurónu M_j^i vezmeme funkciu príslušnosti fuzzy množiny A_{i_j}.

Ako z tohoto príkladu vidieť, pomocou algoritmu 3 dokážeme získať počiatočné nastavenie NFC, ktoré môžeme ďalej zdokonaľovať niektorým z učiacich algoritmov.

5 Predikcia pomocou klasifikátorov

Z matematického hľadiska pod predikciou rozumieme určenie člena x^t nejakej (časovej) postupnosti $\{x^i\}_{i \geq 0}$ reálnych čísel (vo všeobecnosti prvkov nejakého univerza $\mathcal{U}$) pomocou už známych členov x^k, $k = 1, \ldots, t-1$. Predpokladáme pritom, že existuje prirodzené číslo d, nazývané tiež **stupeň predikcie** a funkcia $f : \mathrm{R}^d \to \mathrm{R}$ taká, že

$$x^t = f(x^{t-d}, x^{t-d+1}, \ldots, x^{t-1}).$$

Úlohou je potom nájsť takú funkciu $g : \mathrm{R}^d \to \mathrm{R}$, ktorá dostatočne dobre aproximuje f.

V praxi je však často výhodnejšie predpovedať chovanie nejakého systému pomocou r parametrov $p_1, \ldots, p_r \in \mathrm{R}$, ktoré chovanie systému ovplyvňujú. Úlohou je teda aproximovať nejakú funkciu $f' : \mathrm{R}^{rd} \to \mathrm{R}$ takú, že

$$x^t = f'(p_1^{t-d}, p_1^{t-d+1}, \ldots, p_1^{t-1}, p_2^{t-d}, \ldots, p_2^{t-1}, \ldots, p_r^{t-d}, \ldots, p_r^{t-1}),$$

kde x^t je hodnota nejakej veličiny x charakterizujúcej chovanie systému. Predikcia pomocou neurónových sietí potom znamená zostrojenie takej neurónovej siete, ktorá je dobrou aproximáciou funkcie f' (resp. f).

V prípade, že na základe parametrov $p_1, \ldots, p_r$ nepotrebujeme predpovedať presnú hodnotu x^t veličiny x, ale iba príslušnosť x^t do jednej z m tried $C^1, \ldots, C^m \subseteq \mathrm{R}$, je našou úlohou nájsť aproximáciu takejto funkcie $f'' : \mathrm{R}^{rd} \to \{0,1\}^m$. Pre $\mathbf{u} \in \mathrm{R}^{rd}$ pritom $f''(\mathbf{u}) = \mathbf{e}^k$ znamená, že hodnota x^t veličiny x bude v čase t patriť do triedy C^k. Funkcia f'' je vlastne špeciálnym druhom klasifikačnej funkcie a preto je pre takúto predikciu vhodné použiť neurónové siete pracujúce ako klasifikátory.

6 Geomagnetické búrky

Magnetické pole Zeme je možné považovať za pole veľkého trvalého magnetu. Smer a sila tohoto poľa sa však neustále mení. Niekedy sú zmeny nepatrné, inokedy dochádza k výrazným výkyvom, ktoré trvajú niekoľko hodín a ktoré nazývame geomagnetickými búrkami. Hlavnou príčinou týchto zmien sú procesy prebiehajúce na Slnku. Slnečné erupcie spôsobujú výrony častíc do medziplanetárneho priestoru, čo spôsobuje výkyvy v magnetickom poli Zeme. Prúd týchto častíc slnečného pôvodu nazývame slnečným vetrom.

Na sledovanie zmien magnetického poľa Zeme sa používa veličina D_{st}, nazývaná indexom geomagnetického poľa Zeme. Jej hodnota sa v období kľudu pohybuje v rozmedzí $\pm$20nT, počas magnetickej búrky však môže v priebehu niekoľkých hodín klesnúť až o niekoľko sto nT.

V priebehu geomagnetickej (GM) búrky je možné rozoznávať 3 fázy: počiatočnú fázu, hlavnú fázu a fázu obnovy. V počiatočnej fáze najprv nastáva mierne zvýšenie a neskôr prudký pokles hodnoty D_{st}. V hlavnej fáze Dst dosahuje minimálnu úroveň a vo fáze obnovy sa postupne dostáva na svoju pôvodnú úroveň. Pre nás je najdôležitejšia prvá fáza. Tá je spravidla sprevádzaná zvýšením počtu častíc n slnečného vetra, ako aj výraznou zmenou z-ovej zložky B_z vektora intenzity medziplanetárneho magnetického poľa.

Dáta, ktoré sme mali k dispozícii pre predikciu GM búrok, obsahovali hodnoty nasledujúcich veličín:

- B_z ... z-ová zložka vektora intenzity medziplanetárneho magnetického poľa
- σ_{B_z} ... stredná kvadratická odchýlka B_z, charakterizujúca výkyvy tejto veličiny
- n ... počet častíc slnečného vetra v 1cm^3
- v ... rýchlosť častíc slnečného vetra.

My sme sa výskyt geomagnetickej búrky snažili určiť pomocou dvoch parametrov: σ_{B_z} a nv. Parameter nv pritom predstavuje tlak slnečného vetra na magnetosféru Zeme.

7 Spôsob predikcie

Predikciu GM búrky budeme vykonávať tak, že na základe hodnôt parametrov nv^{t-1} a $\sigma_{B_z}^{t-1}$ sa budeme snažiť určiť, či v časovom intervale $\langle t, t+q\rangle$, pre nejaké prirodzené číslo q, GM búrka nastane, alebo nie. Nepôjde nám teda o to predpovedať hodnotu D_{st} v čase t, ale o to, či v čase t až $t+q$ nastane pokles D_{st} o aspoň 40 nT, alebo nie. Pôjde teda o predikciu tretieho typu so stupňom predikcie 1. Takýto spôsob predikcie je menej bežný, preto sa pokúsime objasniť dôvody, ktoré nás k nemu viedli.

Ako sme už spomenuli, hlavnej fáze GM búrky spravidla predchádzajú viac-menej výrazné zmeny parametrov nv a σ_{B_z}. Výskyt tejto fázy je preto možné popísať nasledovným pravidlom:

Ak nastala výrazná zmena parametrov nv a σ_{B_z}, potom v priebehu najbližších q hodín dôjde k poklesu D_{st} o aspoň 40 nT.

To umožňuje, aby sme na predikciu GM búrok použili neuro–fuzzy klasifikátor. Klasifikovať pritom budeme zmenu ΔD_{st} indexu D_{st} do dvoch tried C_B a C_N, pričom C_B bude trieda poklesov o aspoň 40 nT a C_N bude k nej doplnková trieda, t.j. trieda zmien $\Delta D_{st} > -40nT$.

K výrazným zmenám parametrov nv a σ_{B_z} však nedochádza v presne vymedzenom čase pred hlavnou fázou GM búrky. Tá môže nastať až niekoľko hodín po týchto zmenách, prípadne nemusí nastať vôbec. Je to dôsledkom toho, že n, v, B_z nie sú ani zďaleka všetky parametre, ktoré ovplyvňujú vznik a priebeh GM búrky. Môže sa teda stať, že pri rovnakom priebehu parametrov nv a σ_{B_z} nastane GM búrka v rôznom časovom odstupe, prípadne nenastane vôbec. Parameter q je v našom prípade heuristicky určený parameter, ktorý vyjadruje maximálny počet hodín, o koľko sa môže hlavná fáza búrky opozdiť za časom, v ktorom nastala výrazná zmena parametrov nv a σ_{B_z}. V našej práci sme pracovali s hodnotou $q = 4$.

Jedným z dôvodov, prečo sme sa rozhodli pre stupeň predikcie $d = 1$ je to, že spomínané zmeny parametrov nv a σ_{B_z} nastávajú náhle, trvajú krátko a nepredchádza im žiadna badateľná *prípravná fáza*, ktorá by bola predzvesťou týchto zmien. Hodnoty parametrov nv a σ_{B_z} pred ich výraznou zmenou sa pohybujú v oblasti *bežného šumu*. Druhým dôvodom bolo to, že dáta, ktoré sme mali k dispozícii, obsahovali značné množstvo hodín, v ktorých neboli niektoré z parametrov namerané. Tento fakt značne

obmedzuje činnosť klasifikátora, ktorý sa rozhoduje práve na základe výrazných zmien týchto parametrov. Nahradzovať chýbajúce údaje napríklad interpoláciami nie je v tomto prípade výhodné, nakoľko takýmto spôsobom nenahradíme výkyvy, ktoré mohli v nahradzovanom úseku existovať. Pri znižovaní hodnoty stupňa predikcie d teda zvyšujeme šancu, že hodnoty potrebných parametrov budú v časoch $t-d$ až $t-1$ namerané. Tým aj zvyšujeme možnosť použitia klasifikátora.

8 Vytvorenie klasifikačnej funkcie

Vzhľadom na to, že predpovedanie GM búrok bude v našom prípade vlastne klasifikácia do dvoch tried tvoriacich rozklad reálnych čísel, potrebujeme nájsť aproximáciu nejakej klasifikačnej funkcie

$$f : \mathrm{R} \times \mathrm{R} \to \{0, 1\}.$$

Pritom výsledok $f(nv^{t-1}, \sigma_{B_z}^{t-1}) = 1$ znamená, že v čase $\langle t, t+q\rangle$ nastane GM búrka a výsledok $f(nv^{t-1}, \sigma_{B_z}^{t-1}) = 0$ znamená, že v tomto časovom intervale GM búrka nenastane.

Na tomto mieste je potrebné uvedomiť si, že vo všeobecnosti f nemusí byť funkcia, t.j. môže predstavovať nejednoznačné zobrazenie. To záleží jednak od voľby parametra q, ako aj od množstva iných vonkajších parametrov ovplyvňujúcich priebeh GM búrky. My však pre vytvorenie trénujúcich a testovacích príkladov potrebujeme z dostupných dát vytvoriť iba nejakú diskrétnu funkciu φ. Tá už s najväčšou pravdepodobnosťou bude skutočne funkciou. Ak by sa však predsa len vyskytli drobné nejednoznačnosti v zobrazení φ, spoľahneme sa na schopnosť neurónových sietí vysporiadať sa s chybovými, protirečivými, dátami.

Diskrétnu klasifikačnú funkciu φ teda pomocou dostupných hodnôt parametrov n, v, σ_{B_z} a D_{st} vytvoríme nasledovne:

$$\varphi(nv^{t-1}, \sigma_{B_z}^{t-1}) = \begin{cases} 1 & \text{ak } \min_{t'=t,\ldots,t+q}\{D_{st}^{t'} - D_{st}^{t'-2}\} \leq -40 \\ 0 & \text{inak.} \end{cases} \tag{6}$$

Túto funkciu využijeme na vytvorenie trénujúcej a testovacej vzorky príkladov.

9 Príprava trénujúcich a testovacích dát

Na prípravu trénujúcich a testovacích vzoriek sme použili dáta z rokov 1980 - 1984 a z rokov 1989 - 1992, ktoré obsahovali hodnoty parametrov n, v, B_z, σ_{B_z} a Dst, merané v týchto rokoch každú hodinu. Dôvodom pre výber práve týchto dát bolo to, že obsahujú pomerne málo chýbajúcich, nenameraných, údajov.

Tieto dáta sme najprv spracovali tak, že ak v niektorej hodine chýbala hodnota jedného z parametrov, vypustili sme všetky údaje namerané v tejto hodine z dát. Takto spracované dáta predstavovali postupnosť pätíc $\mathbf{p}^t = (n^t, v^t, B_z^t, \sigma_{B_z}^t, D_{st}^t)$, pričom údaje $\mathbf{p}^t$ a $\mathbf{p}^{t+1}$ nemuseli byť namerané s 1-hodinovým rozdielom.

V druhom kroku sme k takto získanej postupnosti $\{\mathbf{p}^t\}_{t=1}^N$ vytvorili príklady v tvare:

$$(nv^t, \sigma_{B_z}^t, \varphi(nv^t, \sigma_{B_z}^t)), \quad t = 3, \ldots, N-q,$$

kde φ je klasifikačná funkcia (6).

Z príkladov, ktoré sme takýmto spôsobom vytvorili z dát z rokov 1980, 1981 a 1991, sme vytvorili trénujúcu vzorku. Ostatné príklady sme začlenili do testovacej vzorky.

V prípade, ak v časovom intervale dĺžky q pred hlavnou fázou nejakej búrky boli k dispozícii hodnoty všetkých parametrov, tak všetkých q príkladov vytvorených z týchto údajov má tretiu zložku 1. Vzorky sú teda vlastne zložené z *pásov* príkladov, ktorých tretie zložky sú 0 a z (kratších) *pásov* tých príkladov, ktorých tretie zložky sú 1. Jeden takýto *jednotkový* pás príkladov spravidla odpovedá jednej GM búrke. (Niekedy však v rámci jednej GM búrky môže nastať v priebehu niekoľkých hodín aj viacero poklesov. Ak je čas medzi týmito poklesmi väčší ako q, vznikne nám z jednej GM búrky viacero pásov príkladov, s treťou zložkou 1.) Túto vlastnosť trénujúcich aj testovacích vzoriek budeme využívať pri hodnotení úspešnosti klasifikátora.

10 Popis modelu NFC

Na predikciu GM búrok sme použili NFC, ktorý mal dva neuróny vo vstupnej vrstve a jeden neurón vo výstupnej vrstve. Počet neurónov vo fuzzifikačnej a inferenčnej vrstve bol získaný algoritmom 3. Na základe výsledkov tohoto algoritmu sme tiež určili prepojenia medzi prvou a druhou a druhou a treťou vrstvou, ako aj počiatočné parametre výstupných funkcií neurónov vo fuzzifikačnej (druhej) vrstve. Tieto funkie boli trojuholníkového tvaru. Agregačnou funkciou všetkých neurónov inferenčnej vrstvy bola funkcia min. Agregačnou funkciou výstupného neurónu c bola funkcia

$$A_c(o_R) = \max_{R \in N_3} o_R,$$

kde N_3 je množina neurónov tretej vrstvy a o_R je výstup neurónu $R \in N_3$. Výstupná funkcia O_c výstupného neurónu mala tvar

$$O_c(a_c) = \begin{cases} 1 & a_c > 0 \\ 0 & a_c = 0, \end{cases}$$

kde a_c je výsledok agregačnej funkcie tohoto neurónu. Všetky ostatné funkcie vyskytujúce sa v sieti boli identity.

11 Inicializácia siete

Na inicializáciu siete sme využili algoritmus 3. Ten sme sa pokúšali aplikovať na trénujúcu vzorku pri rôznom nastavení jeho parametrov, pričom ako podmienku $\mathcal{P}$ sme uvažovali dvojicu vzťahov (5), (4). Podmienku (4) sme vzhľadom na existenciu iba dvoch tried C_B a C_N upravili nasledovne:

$$\frac{S^i[2]}{S^i[1]} \geq \alpha, \tag{7}$$

kde triede C_N odpovedá jednotkový vektor $\mathbf{e}^1 = (1,0)$ a triede C_B odpovedá vektor $\mathbf{e}^2 = (0,1)$.

Parameter MD určujúci maximálnu povolenú hĺbku vnorenia sme nastavili pevne na hodnotu 7. Rovnako sme zvolili za pevný tvar funkcií príslušnosti. Všetky fuzzy množiny, vyskytujúce sa v predpokladových častiach pravidiel, boli trojuholníkového tvaru s funkciami príslušnosti

$$\mu(x) = \begin{cases} \frac{x-u}{v-u} & x \in \langle u, v\rangle \\ \frac{w-x}{w-v} & x \in (v, w\rangle \\ 0 & \text{inak.} \end{cases}$$

Algoritmus sme potom testovali pri nasledovnom nastavení parametrov: $\alpha = 0.7$, 0.5, 0.35, 0.3, 0.25, 0.2, 0.1; $\beta = 10$, 20, 30; $m_i = 3$, 5, 7 a $b = 3$, 5, 7.

Nastavenie parametra $m_i = 3$ a $m_i = 5$ spôsobovalo, že sa proces nahradzovania fuzzy množín novými a následného znovuvytvárania pravidiel vôbec nevykonal, alebo sa musel opakovať mnohokrát. Často by bolo potrebných viac ako 7 vnorení. Pracovali sme preto hlavne s hodnotou $m_i = 7$.

Pre parameter b sme zvolili hodnotu 3. Nastavenie $b = 7$ spôbovalo to, že sa na jednej úrovni vnorenia vytváralo zbytočne veľa fuzzy množín. Dôsledkom toho bolo vytváranie aj takých pravidiel, ktoré by sa dali zlúčiť do jedného.

Vyššie hodnoty parametrov α a β znižovali počet potrebných nahradzovaní fuzzy množín novými. Viedli tak síce k vytváraniu menšieho počtu pravidiel, avšak pre malý počet fuzzy množín nebolo možné výsledky siete zdokonaliť učením. Najlepšie výsledky sieť vykazovala pri hodnotách $\alpha = 0.3$ a $\beta = 20$.

12 Učiaci algoritmus

Úlohou klasifikátora je nastaviť parametre výstupných funkcií neurónov vo fuzzifikačnej vrstve tak, aby počet nesprávne klasifikovaných príkladov bol čo najmenší, t.j. vytvoriť čo možno najlepšie pokrytie priestorov parametrov fuzzy množinami. Učiaci algoritmus, ktorý sme použili my, vychádza z [5].

Predpokladajme, že vstupnými signálmi do NFC sú parametre nv a σ_{B_z}. Tieto sa spropagujú cez klasifikátor a určí sa jeho výstup $TO \in \{0, 1\}$. Ak $TO = 1$, znamená to, že v súbore pravidiel existuje také, ktoré *zareagovalo* na vstupné signály. V opačnom prípade žiadne takéto pravidlo neexistuje. Označme ďalej AO očakávaný výstup siete, t.j. $AO = \varphi(nv, \sigma_{B_z})$. V závislosti na hodnotách TO a AO môžu nastať 3 prípady:

Ak $AO = TO$, je všetko v poriadku a nie je potrebné parametre funkcií príslušnosti zmeniť.

Ak $AO = 0$ a $TO = 1$, je potrebné parametre niektorých funkcií príslušnosti zmeniť. To je možné dosiahnuť dvoma spôsobmi. Pri prvom spôsobe zmeníme parametre všetkých funkcií príslušnosti, ktoré sa vyskytujú v predpokladovej časti pravidla, ktorého výstup bol kladný, t.j. zmeníme parametre výstupných funkcií všetkých tých neurónov v druhej vrstve, ktoré sú spojené s tými neurónmi v tretej vrstve, ktorých výstup bol kladný. Pri tomto spôsobe je potrebné zabezpečiť, aby sme parametre jednotlivých funkcií príslušnosti adaptovali nanajvýš raz. (Jedna fuzzy množina sa môže vyskytovať v predpokladovej časti viacerých pravidiel.) Druhý spôsob spočíva v tom, že sa najprv určí neurón n_{MSR} odpovedajúci najvýznamnejšiemu pravidlu, t.j. taký, ktorého výstup bol maximálny. Potom sa určí neurón n_{CS} v druhej vrstve klasifikátora, ktorý odpovedá kritickej fuzzy množine najvýznamnejšieho pravidla, t.j. taký, ktorý má najmenší výstup spomedzi všetkých neurnov druhej vrstvy spojených s neurónom n_{MSR}. Adaptujú sa potom iba parametre výstupnej funkcie tohoto neurónu n_{CS}. My budeme v učiacom algoritme pre NFC používať takýto spôsob adaptácie funkcií príslušnosti.

Treťou možnosťou je, že $AO = 1$ a $TO = 0$. V tomto prípade očakávame kladný výstup zo siete, avšak žiadne pravidlo zo súboru pravidiel na vstupné signály nezareagovalo. Nie je teda jasné, parametre ktorých funkcií príslušnosti je potrebné zmeniť. Možností je aj v tomto prípade niekoľko. My v našom učiacom algoritme v takomto prípade nebudeme adaptovať žiadnu fuzzy množinu. V stručnosti objasníme prečo. Spôsob, akým sme vytvárali klasifikačnú funkciu, má zákonite za následok to, že v trénujúcej vzorke budú aj príklady, ktorých hodnoty parametrov nv a σ_{B_z} budú odpovedať bežnému šumu, ale hodnota klasifikačnej funkcie φ bude 1. Takéto príklady vzniknú z hodnôt nameraných v časovom intervale q hodín pred hlavnou fázou GM búrky a vynútia si vznik takejto situácie. Túto chybu preto budeme ignorovať a nebudeme adaptovať parametre žiadnych funkcií príslušnosti.

Učiaci algoritmus pre náš klasifikátor teda vyzerá nasledovne:

Krok 1: Vyber ďalší príklad $(\mathbf{x}^k, AO^k)$ z trénujúcej vzorky.

Krok 2: Vstupný vektor $\mathbf{x}^k$ propaguj cez jednotlivé vrstvy a urči výstup siete TO.

Krok 3: Ak $AO = 0$ a zároveň $TO = 1$, tak
a) $p = (AO - a_c) = -a_c$, kde a_c je výsledok agregačnej funkcie neurónu c z výstupnej vrstvy.

b) Nájdi neurón $R_{MSR} \in N_3$ taký, že $o_{R_{MSR}} = \max_{R \in N_3} \{o_R\}$,
pričom o_R predstavuje výstup neurónu R z tretej vrstvy N_3.

c) Nájdi neurón $M_{CS} \in N_2$, pre ktorý platí
$o_{M_{CS}} = \min_{M \in P} \{o_M\}, \quad P = \{N \in N_2 \mid W(N, R_{MSR}) = 1\}$,
kde o_M je výstup neurónu M z druhej vrstvy N_2.

d) Pre výstupnú funkciu neurónu M_{CS} urči delta hodnoty jej
parametrov u, v, w použijúc učiaci pomer $\eta > 0$:

$\delta_v = \eta.p.(w - u).\mathrm{sgn}(x_i - v)$,
$\delta_u = -\eta.p.(w - u) + \delta_v$,
$\delta_w = \eta.p.(w - u) + \delta_v$,

kde x_i odpovedá neurónu N_i vstupnej vrstvy takému, že $W(N_i, M_{CS}) = 1$.
Adaptuj jednotlivé parametre:

$u = u + \delta_u$,
$v = v + \delta_v$,
$w = w + \delta_w$.

Inak pokračuj krokom 4.

Krok 4: Ak je koniec učiacej epochy a je splnené kritérium ukončenia skonči.
Inak pokračuj krokom 1.

Väčšinou sme pracovali s hodnotou učiaceho pomeru $\eta = 0.1$ a nasledovným kritériom ukončenia učiaceho procesu: Učiaci proces sme zastavili, ak k zmenám parametrov u, v, w, v kroku 3 došlo počas jedného učiaceho cyklu v menej ako 1% príkladov z trénujúcej vzorky.

13 Spôsob hodnotenia a výsledky

Úspešnosť činnosti klasifikátora sme vyhodnocovali pomocou troch testovacích vzoriek vytvorených z dát jednotlivých rokov. Kritériom (ne)úspešnosti pritom boli dve hodnoty E_1 a E_2, predstavujúce chybu prvého a druhého druhu. Význam symbolov použitých pri ich definovaní je nasledujúci:

- B ... skutočný počet GM búrok v príslušnej vzorke. Táto hodnota by odpovedala počtu pásov príkladov, ktorých tretia zložka je 1, v prípade, keby sme neodstraňovali príklady s chýbajúcimi údajmi.
- S ... počet búrok správne predpovedaných sieťou. Za úspešne predpovedanú búrku pritom považujeme situáciu, keď výstup siete je pre aspoň jeden príklad z *jednotkového pásu rovný 1.*
- n ... *počet príkladov testovacej vzorky, ktorých tretia zložka sa rovná 0.*
- M ... *počet takých príkladov, pri ktorých sieť dala výstup 1, ale tretia zložka týchto príkladov je 0. Toto číslo zodpovedá počtu búrok, ktorých výskyt sieť predpovedala, aj keď tieto v skutočnosti nenastali.*

Chyba prvého druhu

$$E_1 = 1 - \frac{S}{B},$$

predstavuje neúspešnosť siete pri predpovedaní búrok, ktoré nastali. Chyba druhého druhu

$$E_2 = \frac{M}{n}.$$

zasa vyjadruje, v akej časti príkladov sieť predpovedala búrku, aj keď táto v skutočnosti nenastala. Učiaci algoritmus uvedený v predchádzajúcej stati je zameraný na minimalizáciu tejto chyby.

Ako sme uviedli, testovaciu vzorku sme vytvorili z dát z rokov 1982-1984, 1989, 1990 a 1992. Dáta z rokov 1982-1984 predstavovali kompletné údaje merané každú hodinu počas týchto rokov. Dáta z ostaných rokov však neboli úplné. Obsahovali iba údaje namerané 48 hodín pred a 96 hodín po GM búrkach vyskytujúcich sa v období týchto rokov. Naviac, všetky chýbajúce údaje boli nahradené interpoláciami. Z tohoto dôvodu sme vytvorili tri testovacie vzorky príkladov: Vzorka A obsahovala 9975 príkladov vytvorených z rokov 1982-1984. Vzorka B obsahovala 7674 príkladov vytvorených z rokov 1989, 1990 a 1992. Posledná, tretia, vzorka C obsahovala 17649 príkladov z oboch predchádzajúcich vzoriek. Vzorka A pritom obsahovala 32, vzorka B 94 a vzorka C 126 GM búrok (presnejšie pásov príkladov, ktorých tretia zložka je 1).

Tri najlepšie výsledky, ktoré sme pri klasifikácii dosiahli, uvádzame v tabuľke. Symbol R predstavuje počet pravidiel použitých pri klasifikácii a S počet správne predpovedaných búrok. Namiesto chyby prvého druhu E_1 uvádzame v tabuľke údaje $1 - E_1$, ktoré vyjadrujú úspešnosť klasifikátora pri predikcii GM búrok. Oba údaje $1 - E_1$ a E_2 sú uvádzané v percentách.

			Vzorka A ($B = 32$)			Vzorka B ($B = 94$)			Vzorka C ($B = 126$)		
α	β	R	S	$1 - E_1$	E_2	S	$1 - E_1$	E_2	S	$1 - E_1$	E_2
0.3	20	74	17	53.12	0.91	33	35.11	1.56	50	39.68	1.18
0.35	20	75	16	50.00	0.97	29	30.85	1.86	45	35.71	1.34
0.25	20	100	16	50.00	1.08	26	27.66	2.20	42	33.33	1.55

Tabuľka 2: Výsledky získané neuro–fuzzy klasifikátorom po skončení učiaceho procesu.

Z tabuľky 2 je vidieť výrazný rozdiel vo výsledkoch pre vzorku A a vzorku B, ktorý je s najväčšou pravdepodobnosťou spôsobený nahradzovaním chýbajúcich údajov vo vzorke B interpoláciami.

Zaujímavé na výsledkoch uvedených v tabuľke 2 je to, že boli získané po jednej až dvoch učiacich epochách. To svedčí o schopnosti algoritmu 3 urýchliť učiaci proces siete vytvorením jej vhodnej štruktúry a počiatočným nastavením jej parametrov na základe analýzy trénujúcej vzorky.

			Vzorka A ($B = 32$)			Vzorka B ($B = 94$)			Vzorka C ($B = 126$)		
α	β	R	S	$1 - E_1$	E_2	S	$1 - E_1$	E_2	S	$1 - E_1$	E_2
0.3	20	74	19	59.38	1.57	36	38.30	2.44	55	43.65	1.93
0.35	20	75	19	59.38	1.78	39	41.49	2.95	58	46.03	2.27
0.25	20	100	17	53.12	2.19	39	41.49	4.40	56	44.44	3.11

Tabuľka 3: Výsledky získané bez učiaceho procesu.

Údaje uvedené v tabuľke 3 sú výsledkom siete inicializovanej pomocou algoritmu 3, u ktorej ešte nebol zahájený učiaci proces. Ak by sme sa teda uspokojili s chybou druhého druhu E_2 okolo dvoch až štyroch percent, bola by funkcia neurónovej siete v tomto NFC zbytočná.

Zaujímavé by bolo porovnanie výsledkov, ktoré sme dosiahli pri predikcii stupňa 1 s výsledkami pri predikcii vyšších stupňov. S počtom vstupných premenných však prudko rastie aj počet pravidiel, ktorými sa klasifikácia riadi. To môže obmedzovať činnosť algoritmu 3 hlavne z pamäťového hľadiska, nakoľko tento algoritmus potrebuje pre správny výber pravidiel informácie o všetkých pravidlách, ktoré je možné z trénujúcich dát vytvoriť.

References

[1] Siegfried Gottwald: *Fuzzy Sets and Fuzzy Logic*, Vieweg, 1993, ISBN 3-528-05311-9

[2] Vilém Novák: *Fuzzy množiny a jejich aplikace,* SNTL Praha, 1986

[3] Petr Hájek: *Fuzzy logic from the logikal point of view,* SOFSEM '95, Springer, LNCS 1012

[4] M. M. Gupta, D. H. Rao: *On the principles of fuzzy neural networks,* Fuzzy Sets and Systems 61, 1994, str. 1–18

[5] Detlef Nauck and Rudolf Kruse: *NEFCLASS — a neuro–fuzzy approach for the classification of data,* článok prednesený na "Symposium on Applied Computing" počas "1995 ACM Computing weak", Nashville, 26.2 – 2.3 1995

[6] Programový systém NEFCLASS

[7] P. Sinčák, G. Andrejková: *Neurónové siete I,* ELFA, Košice, 1996, ISBN 80-88786-38-X

[8] P. Sinčák, G. Andrejková: *Neurónové siete II,* ELFA, Košice, 1996, ISBN 80-88786-42-X