Rozhodovacie džungle a iné klasifikačné úlohy dolovania údajov

- **riešiteľ:** Bc. Šimon Horvát
- **vedúci práce:** RNDr. Ľubomír Antoni, PhD.
- **kontakt:** simon.h1231@gmail.com

„Žijeme v dobe kedy sa dáta stavajú komoditou“, povedal istý profesor českej univerzity na videokonferencii, ktorej som sa v rámci extrapolácií mal možnosť zúčastniť.

Je teda pravdou, že „sme bohatí na dáta, ale chudobní na informácie” (Tyždeň vedy a techniky, UPJS).

Data mining je oblasť informatiky, ktorá sa zaoberá transformáciou dát na poznatky a znalosti. Na túto „extrakciu poznatkov“ z dát sa používajú *machine learning - algoritmy*, ktorým sa budem vo svojej práci venovať.

Proces objavovania znalostí predkladá mnoho úloh: úlohy **klasifikácie** (áno/nie a pod.), úlohy **predikcie** (numerické atribúty), **zhlukovanie**, asoc. pravidlá a podobne.

Spolu so svojím školiteľom sa budeme zaoberať **klasifikačnými** úlohami. Klasifikáciou nazývame proces zaradenia pozorovaní do jednej z tried.

Označme si triedu $y_i \in \{1, 2 \dots C\}$
Tréningová množina (X) v tomto prípade vyzerá takto:
$X = \{(x_1,y_1) \dots (x_m, y_m)\} \subseteq R^n \times \{1, 2 \dots C\}$
Úlohou klasifikácie je teda nájsť na základe poskytnutých informácií (X), pre každé $x \in R^n$ zodpovedajúce $y \in \{1, 2 \dots C\}$.

Prvým modelom na klasifikáciu, ktorým sa budem zaoberať je **rozhodovací strom.** Rozhodovacie stromy sú jednou z najobľúbenejších (pre jednoduchú interpretáciu) dataminigových techník. Každý **uzol** stromu predstavuje jednu (vybranú) **vlastnosť objektov**, z tohto uzlu vedie konečný počet hrán. Preto je

nutné vlastnosti najskôr **diskretizovať** (napríklad z reálnych čísel do konečného počtu intervalov).

Kameň úrazu je však vo vytváranie takéhoto stromu. Ten musí čo najlepšie jednotlivé objekty od seba odlíšiť. Pre koreňový uzol sa **vyberá taký atribút, ktorý objekty od seba maximálne odlíši**. Využíva sa preto **entropia** (miera informačnej hodnoty atribútu), informačný zisk, **Chí-kvadrant**, **Gini index** atď.

Množstvo vyvinutých algoritmov na učenie sa rozhodovacích stromov je variáciou základného algoritmu. Pôvodná myšlienka konštrukcie rozhodovacích stromov siaha do päťdesiatych rokov dvadsiateho storočia. Konkrétne ide o prácu o koncepčných učiacich sa systémoch (CLS) od Hovelanda a Hunta.

Algoritmus pozostáva z piatich krokov:

1. Do T priraď celú tréningovú množinu. Vytvor T uzol.
2. Ak sú všetky príklady v T pozitívne, vytvor uzol P, ktorý je nasledovníkom T uzla a skonči.
3. Ak sú všetky príklady v T negatívne, vytvor uzol N, ktorý je nasledovníkom T uzla a skonči.
4. Vyber atribút X s hodnotami v_1, v_2, ..., v_N a rozdeľ T do podmnožín T_1, T_2, ..., T_N priradiac ich hodnoty do X. Vytvor N uzlov Ti_i(i = 1, ..., N) ako potomkov uzla T a X = v_i ako hodnotu vetvi od T k T_i.
5. Pre každé T_i rob: do T priraď T_i a choď na krok 2.

Dôležitým kritériom v algoritme rozhodovacieho stromu je výber atribútu na testovanie v každom rozhodovacom uzle stromu. Cieľom je vybrať taký atribút, ktorý najlepšie klasifikuje príklady. Dobré kvantitatívne meranie vhodnosti atribútu poskytuje štatistická vlastnosť nazývaná **informačný zisk**, ktorá udáva mieru do akej atribút rozdeľuje tréningové príklady do ich cieľovej klasifikácie. Na výpočet informačného zisku sa využíva **entropia**. Toto meranie sa uskutočňuje pri výbere z kandidátskych atribútov v každom kroku rastu stromu.

Keďže výber optimálneho atribútu je dôležitý nie len pre stromy, ale aj pre **rozhodovanie lesy** a **džungle**, budeme sa týmito metódam zaoberať viac.

Entropia - miera homogenity množiny príkladov

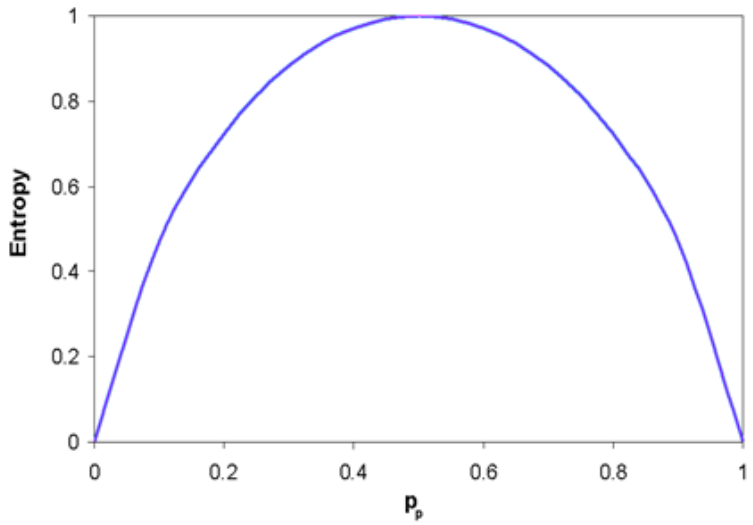
Aby sme precízne stanovili informačný zisk, musíme zadefinovať entropiu, ktorá charakterizuje (ne)čistoty v ľubovoľnej skupine príkladov. Daná je množina S, obsahujúca iba pozitívne a negatívne príklady nejakého cieľového konceptu. Potom entropia množiny S, zodpovedajúcej tomuto jednoduchému príkladu binárnej klasifikácie, je definovaná ako:

$$\textbf{Entropia}(S) = - p_p \log_2 p_p - p_n \log_2 p_n$$

Kde p_p je podiel pozitívnych príkladov v S a p_n je podiel negatívnych príkladov v S.Vo všetkých výpočtoch týkajúcich sa entropie definujeme 0log0 rovný 0.
Na ilustráciu, nech S je zbierka 25 príkladov, kde 15 je pozitívnych a 10 je negatívnych. Potom entropia S zodpovedajúcej tejto klasifikácii je

$$\textbf{Entropia}(S) = - (15/25) \log_2 (15/25) - (10/25) \log_2 (10/25) = 0.970$$

Entropia je rovná 0 ak všetky členy S spadajú do rovnakej triedy.
Napríklad, ak všetky členy sú pozitívne (p_p= 1), potom pn je 0, a
$\textbf{Entropia}(S) = -1*\log_2(1) - 0*\log_2 0 = -1*0 - 0*\log_2 0 = 0$.
Entropia dosahuje maximálnu hodnotu, teda 1, ak množina príkladov obsahuje rovnaký počet pozitívnych aj negatívnych príkladov. Ak množina obsahuje rôzny počet pozitívnych a negatívnych príkladov, entropia sa pohybuje v rozmedzí od 0 po 1. Nasledujúci obrázok ukazuje priebeh funkcie entropie zodpovedajúcej binárnej klasifikácii.



Priebeh funkcie entropie zodpovedajúcej binárnej klasifikácii

Informačný zisk

Pre danú entropiu merajúcu nečistotu v množine tréningových príkladov môžeme definovať mieru efektívnosti atribútu v klasifikácii tréningových údajov. Informačný zisk je vlastne očakávané zmenšenie entropie zapríčinené rozdelením príkladov týkajúcich sa daného atribútu. Presnejšie, informačný zisk **Gain (S, A)** atribútu A, zodpovedajúceho množine príkladov S je definovaný ako

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

kde **Values**(A) je množina všetkých možných hodnôt pre atribút **A** a S_v je podmnožina **S**, pre ktorú atribút a má hodnotu v. Prvým členom rovnice pre výpočet zisku **Gain** je entropia pôvodnej zbierky **S** a druhým je očakávaná hodnota entropie po rozdelení **S** atribútom **A**. Očakávaná entropia opísaná druhým členom rovnice je vlastne suma entropií pre každú podmnožinu S_v, váhovaná podielom príkladov $|S_v|/|S|$ patriacich do S_v. **Gain** (S,A) je preto očakávaným zmenšením entropie spôsobeným známou hodnotou atribútu A. Inak povedané, **Gain**(S,A) je informácia poskytovaná o konkrétnej hodnote atribútu, danej hodnotou iného atribútu A. Hodnota **Gain**(S,A) je počet bitov ušetrených pri zakódovaní konkrétnej hodnoty ľubovolného člena **S** pri známej hodnote atribútu **A**.

Proces výberu nového atribútu a rozdeľovania tréningových príkladov je teraz opakovaný pre každý neterminálny klesajúci uzol, tentokrát s využitím výlučne tréningových príkladov spojených s daným uzlom. Atribúty, ktoré boli zapracované vyššie v strome sú vylúčené, takže každý atribút sa môže objaviť najviac jedenkrát pri ľubovoľnom prechode stromom. Tento proces pokračuje pre každý nový list kým nie je splnená aspoň jedna z dvoch podmienok:

1. každý atribút už bol zaradený do stromu
2. všetky tréningové príklady spojene s daným listom majú spoločnú hodnotu atribútu.

Pri učení sa rozhodovacích stromov sa stretávame s rôznymi problémami. K najbežnejším a najzávažnejším patrí napríklad problém s určovaním hĺbky, do ktorej má rozhodovací strom narásť, spracovanie spojitých atribútov, výber vhodnej selekcie atribútu, spracovanie tréningových dát s chýbajúcimi

hodnotami atribútov, spracovanie atribútov s rozdielnym ohodnotením a zvyšovanie efektívnosti výpočtu.

Medzi silné stránky metód rozhodovacieho stromu patria:

- Rozhodovacie stromy sú schopné generovať pochopiteľné pravidlá.
- Rozhodovacie stromy dosahujú klasifikáciu bez potreby prílišného počítania.
- Rozhodovacie stromy sú schopné pracovať s kontinuálnymi aj kategorickými premennými.
- Rozhodovacie stromy poskytujú jasnú indikácia, ktoré oblasti sú najdôležitejšie pre predikciu alebo klasifikáciu.

Medzi slabé stránky patrí:

- Rozhodovacie stromy sú menej vhodné pre výpočet úloh, kde cieľom je predikcia hodnôt kontinuálneho atribútu.
- Rozhodovacie stromy sú náchylné k chybám v klasifikácii problémov s mnohými triedami a relatívne malým počtom tréningových príkladov.
- Rozhodovacie stromy môžu byť výpočtovo náročné na trénovanie. Proces rastu rozhodujúceho stromu je výpočtovo náročný. Na každom uzle, každý kandidát na rozdelenie musí byť usporiadaný predtým ako sa nájde jeho najlepšie rozdelenie.

Ak nepoužijeme žiadne podmienky na orezávanie, vznikne veľmi veľký strom. Model sa tzv. preučí – bude fungovať výborne na tréningovej množine, ale na testovacej zlyhá.

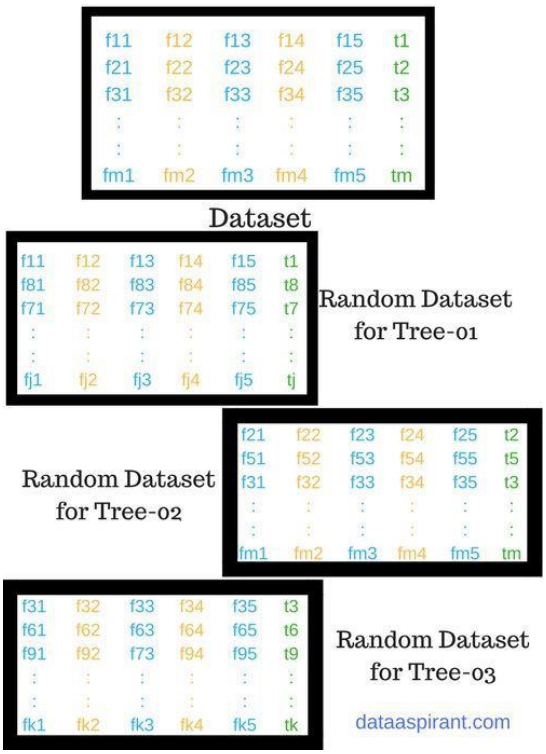
Tento problém preučenia riešia náhodné rozhodovacie lesy. Fungujú na princípe generovania veľkého počtu náhodných rozhodovacích stromov a následného zagregovania ich výsledkov.

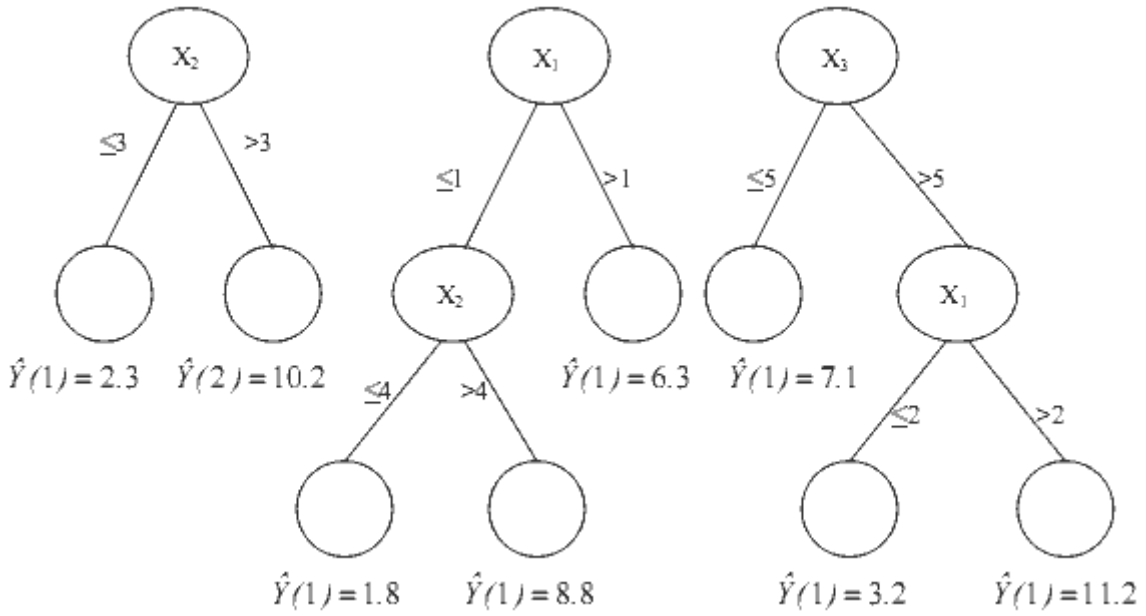
Výhodou týchto stromov je tiež to, že jednotlivé stromy na sebe nezávisia, a teda je možný paralelný výpočet.

Sú to sady rozhodovacích stromov, ktoré boli vytvorené pomocou rôznych metód alebo rovnakou metódou (rôzne parametre). Tieto sady nám pomáhajú vylepšiť kvalitu predpovede a umožnia lepšie pochopenie zákonov skúmaného problému.

Uvažujme sadu stromov a skúmanie problému **x**. Každý strom nám dá na problém svoju odpoveď. Ako však nájsť všeobecnú (spoločnú) odpoveď pre predpovedanú hodnotu **Y**? Najľahšou cestou je získať odpoveď od všetkých stromov. Je to buď voľba správnej hodnoty hlasovaním (väčšina zvíťazí), alebo spriemerovaním.

Ako príklad je možné uviesť sadu stromov na nasledujúcom obrázku. Za x si zvolíme hodnoty (3,4,8). Spoločný výsledok bude **Y**(x)=(10.2+6.3+11.2)/3 = 9.233.





Algoritmus tvorby lesa môžeme popísať nasledovne:

1. Vytvor bootstrapový podsúbor L_i o veľkosti N - trénovací súbor.

2. Vyber náhodne m predátorov.

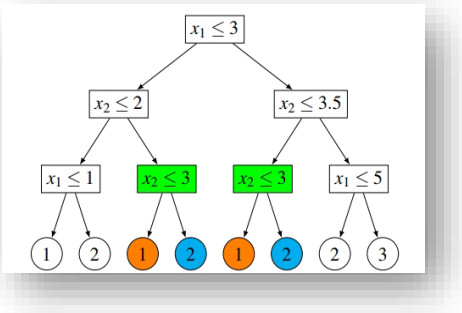
3. Vytvor strom T_i na súboru Li iba s použitím m náhodne vybraných predátorov. Rast stromov sa zastaví, až strom dosiahne minimálne hodnoty veľkosti uzlov.

4. Zaraď *oob* pozorovaní (testovací súbor) vytvoreným stromom a urči výslednú klasifikačnú triedu (kategóriu) alebo predikciu všetkých *oob* pozorovaní.

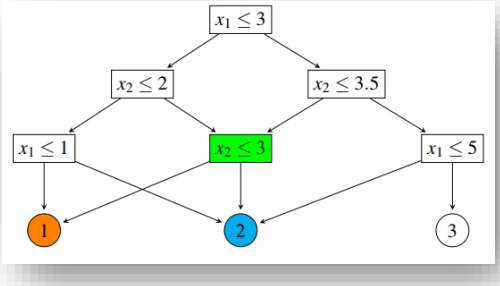
Krok 1-4 sa opakuje do konečného počtu stromov v lese.

5. Spočítaj celkový výsledok klasifikácie/predikcie celého lesa väčšinovým hlasovaním/priemerovaním.

Nevýhodou náhodných lesov je však požiadavka na pamäť, ktorá rastie exponenciálne s hĺbkou stromov. Tento problém by však mal riešiť nový model – Rozhodovacie džungle.



Rozhodovací strom



Rozhodovací DAG

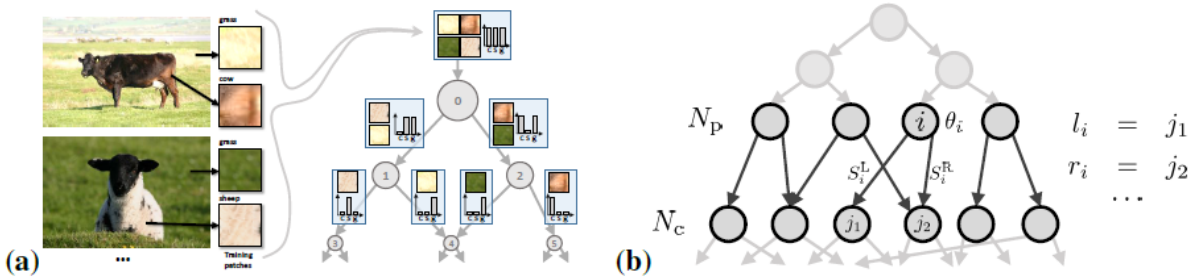
S využitím orientovaných hrán, táto štruktúra poskytuje rôzne cesty od „koreňa“ ku konkrétnemu „listu“, čo by malo prispieť k zlepšeniu pamäťových nárokov.

Tak ako rozhodovací les je sada rozhodovacích stromov, tak isto je rozhodovacia džungľa je sada DAG-ov (directed acyclic graph).

definícia Rozhodovací DAG je orientovaný acyklický graf G = (V,E) s nasledujúcimi vlastnosťami. Nelistový uzol *v* je reprezentovaný :

- Feature dimension d_v z {1, .., n}
- prah θ_v z R.
- v_l ľavý uzol z V
- r_l pravý uzol z V

príklad



(a) Príklad používania DAG na klasifikáciu:

obrazovkových škvŕn patriace do tried **tráva**, **krava** alebo **ovca**. Použitie DAG namiesto stromov znižuje počet uzlov a môže viesť k lepšej generalizácii. Napríklad rôzne farebné škvrny trávy (žltá a zelená) sú zlúčené do uzla 4, pretože to sú podobné štatistiky triedy. Toto môže podporiť zovšeobecnenie tým, že reprezentuje skutočnosť, že tráva sa môže vyskytovať ako zmes žltej a zelenej farby. **(b)** Označenie pre DAG, jeho uzly, znaky a vetvy.

Každý DAG trénujeme nezávisle v džungli. Metóda učenia DAG funguje tak, že v každom kroku rastie o jednu úroveň.

Učenie alebo trénovanie rozhodovacieho DAG sa týka úlohy nájsť **optimálne parametre** (d_v, θ_v, l_v, r_v) pre každý uzol **v**. Učenie rozhodovacieho DAG je v podstate náročnejšie než sa tréning binárneho rozhodovacieho stromu, pretože parametre l_v a r_v (t.j. štruktúra grafov) sú fixné pre binárne stromy.

Binary decision trees

At each node v optimize

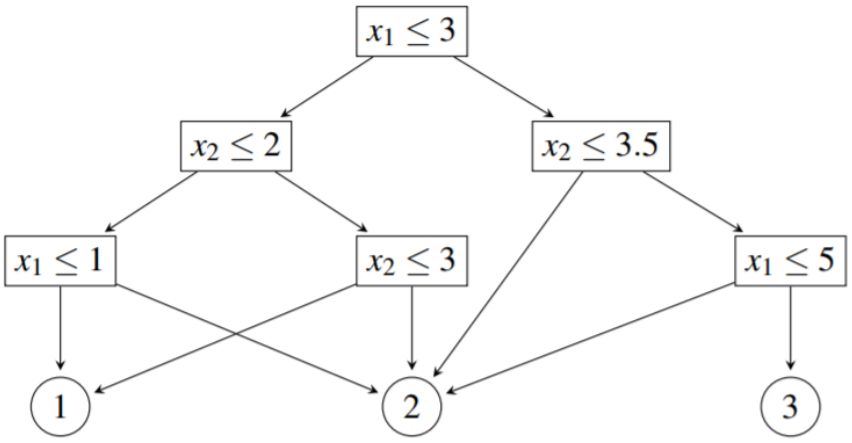
- the feature d_v
- the threshold θ_v

Decision DAGs

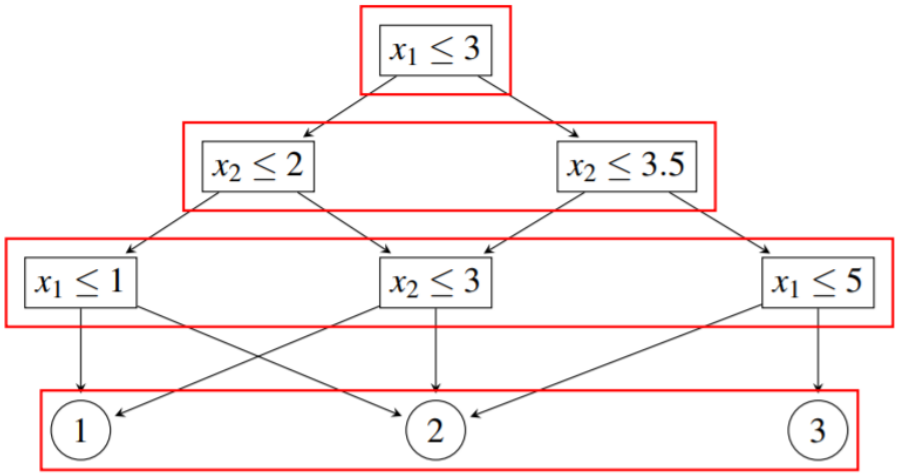
At each node v optimize

- the feature d_v
- the threshold θ_v
- the left child node l_v
- the right child node r_v

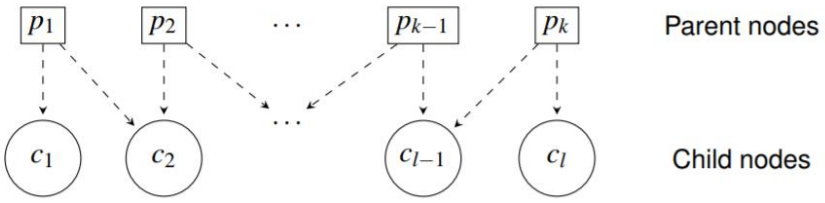
Technicky brané, na nasledujúcom obrázku je tiež DAG.



Avšak, v práci budeme predpokladať/pracovať s pravidelnejším typom tejto štruktúry:

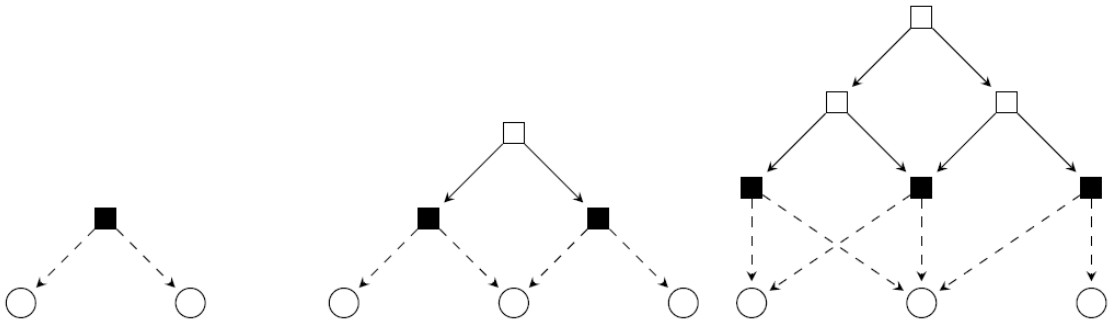


V nasledujúcich státiach budeme používať toto označenie:



- feature dimension d_{p_i}
- threshold θ_{p_i}
- left child node l_{p_i}
- right child node r_{p_i}
- S_{p_i} and S_{c_j} are the training sets at nodes p_i and c_j respectively

Trénovanie DAG-ov ma na starosti LSearch algoritmus, ktorý trénuje v jednom kroku jednu úroveň grafovej štruktúry.



(a) First iteration (b) Second iteration (c) Third iteration

Pri každej iterácii algoritmu sa do grafu pridá nová úroveň uzlov a určia sa atribúty predchádzajúcich listov. Vyššie uvedený obrázok zobrazuje prvé tri iterácie.
Štvorcové uzly predstavujú vnútorné uzly a okrúhle uzly predstavujú uzly listov. Plnené uzly sú tie, ktorých parametre sa pravé určujú.

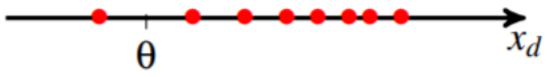
Algorithm 1 LSEARCH

1: **function** LSEARCH($\Theta_{p_1}, ..., \Theta_{p_k}$)
2: **while** something changes **do**
3: **for** $i = 1, ..., k$ **do**
4: $\mathcal{F} \leftarrow$ random feature selection
5: $(d_{p_i}, \theta_{p_i}) \leftarrow \arg\min_{d \in \mathcal{F}, \theta \in R} E(\Theta_{p_1}, ..., \Theta_{p_{i-1}}, (d, \theta, l_{p_i}, r_{p_i}), \Theta_{p_{i+1}}, ..., \Theta_{p_k})$
6: **end for**
7: **for** $i = 1, ..., k$ **do**
8: $l_{p_i} \leftarrow \arg\min_{l=c_1,...,c_l} E(\Theta_{p_1}, ..., \Theta_{p_{i-1}}, (d_{p_i}, \theta_{p_i}, l, r_{p_i}), \Theta_{p_{i+1}}, ..., \Theta_{p_k})$
9: $r_{p_i} \leftarrow \arg\min_{r=c_1,...,c_l} E(\Theta_{p_1}, ..., \Theta_{p_{i-1}}, (d_{p_i}, \theta_{p_i}, l_{p_i}, r), \Theta_{p_{i+1}}, ..., \Theta_{p_k})$
10: **end for**
11: **end while**
12: **return** $\Theta_{p_1}, ..., \Theta_{p_k}$
13: **end function**

Na optimalizáciu chybovej funkcie **E** (riadok 5) je možne použiť entropiu/informačný zisk (metódy vysvetlené vyššie).

Testovanie viacerých prahov pre fixnú dimenziu vektorov sady X efektívne

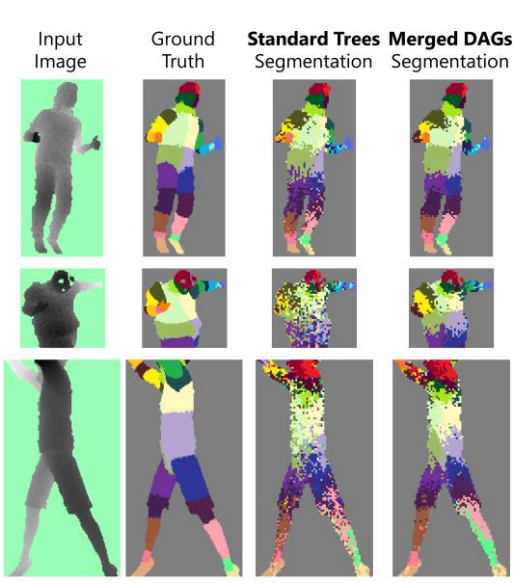
- Zoradiť tréningovú sadu podľa dimenzie vektora x_d
- Následne skúšať prahy medzi susednými bodmi

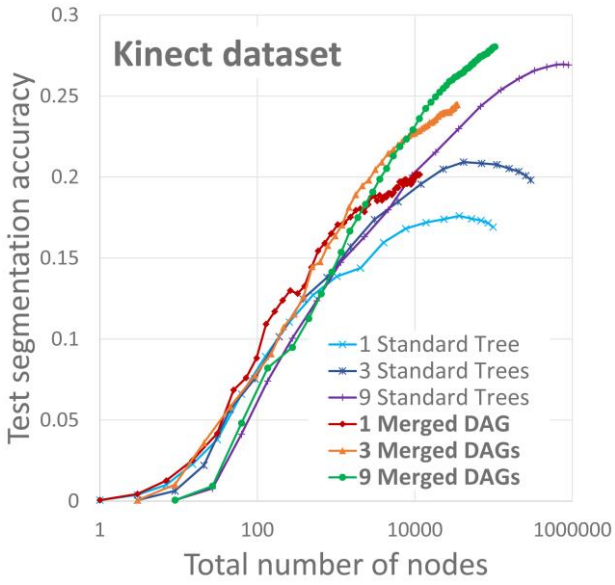


Algorithm 3 Preliminary form of the threshold and feature optimization algorithm. $\mathcal{F}$ is a random feature selection.

for $d \in \mathcal{F}$ **do**
  Sort S_{p_i} according to d-the feature dimension
  for $j = 1, \ldots, |S_{p_i}| - 1$ **do**
    $\theta \leftarrow \frac{(\mathbf{x}_j)_d + (\mathbf{x}_{j+1})_d}{2}$
    if $E(\Theta_{p_1}, \ldots, \Theta_{p_{i-1}}, (d, \theta, l_{p_i}, r_{p_i}), \Theta_{p_{i+1}}, \ldots, \Theta_{p_k}) < E(\Theta_{p_1}, \ldots, \Theta_{p_k})$ **then**
      $d_{p_i} \leftarrow d$
      $\theta_{p_i} \leftarrow \theta$
    end if
  end for
end for

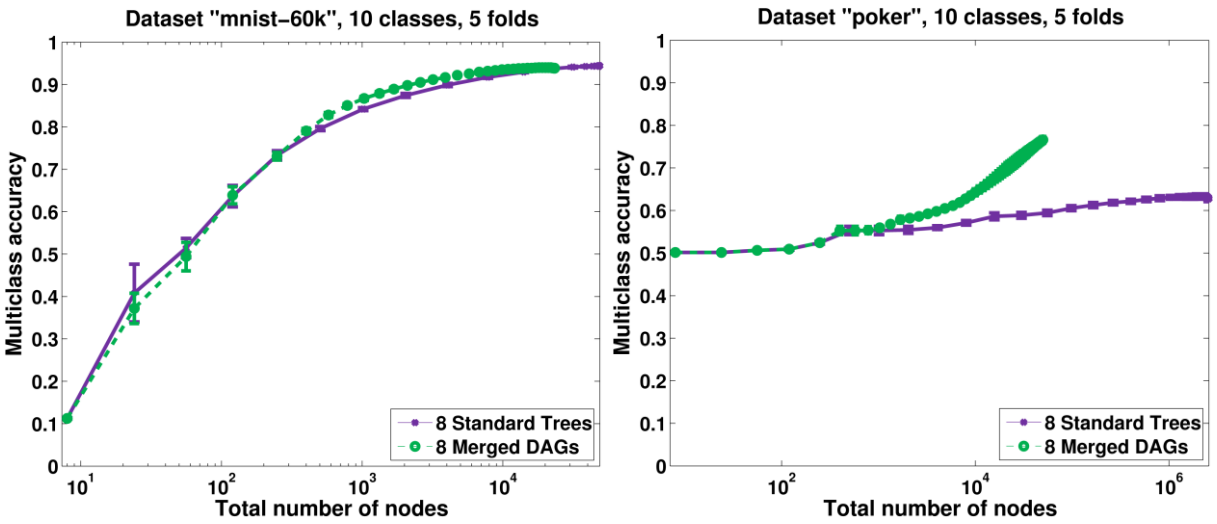
Výsledky uvedené v hlavnom článku (Compact and Rich Models for Classification)



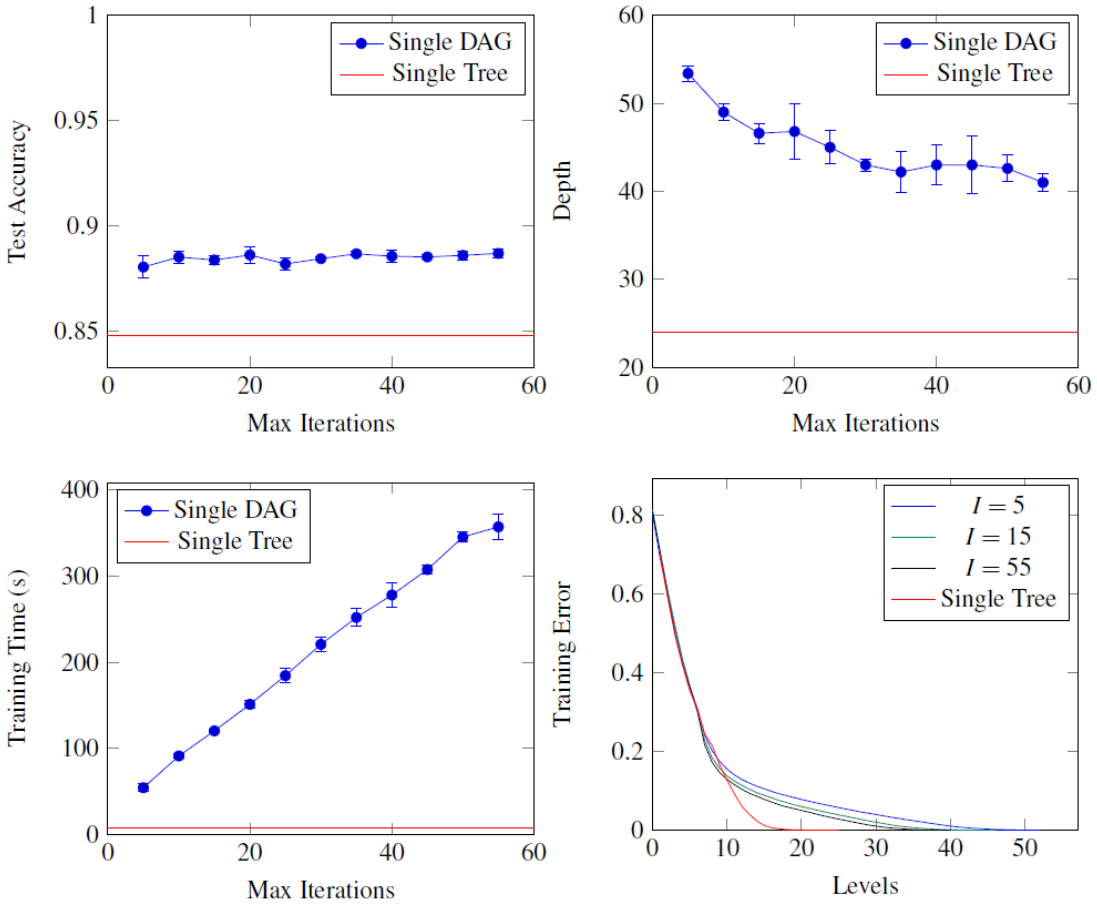


Výsledky získané pre úlohu sémantickej segmentácie na datasete KINECT.
Ľavé obrázky zobrazujú kvalitatívne výsledky, zatiaľ čo pravý graf zobrazuje kvantitatívne výsledky. Zdroj: Shotton et al.

- pamäťové nároky
  Jungle – 3000 uzlov (9MB)
  Forest – 22000 uzlov (80MB)



Výsledky získané pri úlohe klasifikácie v súboroch údajov MINST a POKER.
Zdroj obrázku: Shotton et al.



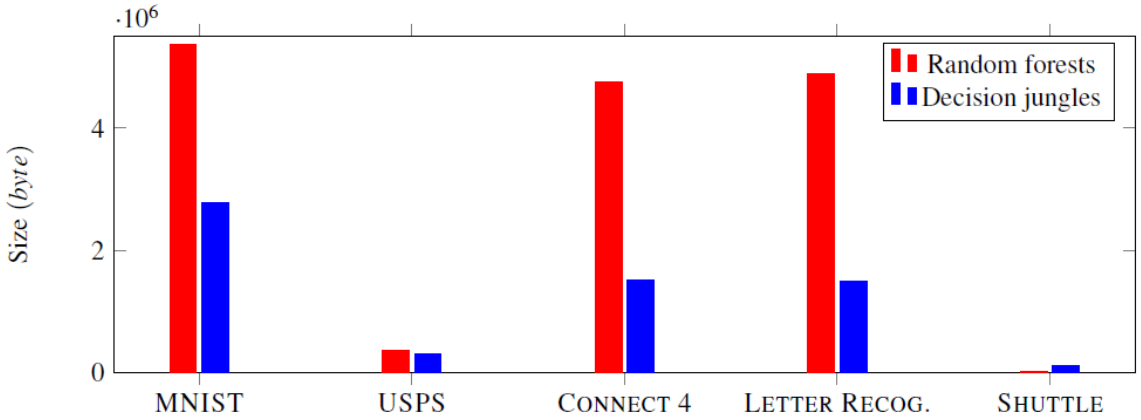
Obrázky ukazujú vplyv maximálneho počtu opakovaní na rôzne výstupné merania.

Data set	Decision jungles				Random forests			
	8 DAGs		15 DAGs		8 Trees		15 Trees	
	Mean	Stdev.	Mean	Stdev.	Mean	Stdev.	Mean	Stdev.
MNIST	**95.72%**	0.13%	**96.38%**	0.09%	95.14%	0.20%	96.23%	0.16%
USPS	**94.65%**	0.5%	**95.95%**	0.2%	94.44%	0.30%	95.93%	0.52%
CONNECT 4	**81.17%**	0.22%	**81.98%**	0.15%	80.99%	0.46%	81.47%	0.66%
LETTER RECOG.	**94.73%**	0.57%	**95.73%**	0.55%	94.29%	0.43%	95.58%	0.48%
SHUTTLE	99.98%	0.01%	**99.99%**	0.01%	**99.99%**	0.00%	**99.99%**	0.01%

Table 3: The table lists the means and standard deviations of the test accuracies.

Data set	Decision jungles (15 DAGs)			Random forests (15 Trees)		
	#internal	#leaf	Size (*byte*)	#internal	#leaf	Size (*byte*)
MNIST	134,936	1,877.2	**2,773,808**	95,940.4	95,955.4	5,373,262.4
USPS	13,113.4	1,012	**302,748**	6,433.2	6,448.2	360,859.2
CONNECT 4	74,633	1,920	**1,515,700**	169,604.4	169,619.4	4,749,103.2
LETTER RECOG.	69,485.6	1,075.2	**1,501,532.8**	40,776.2	40,791.2	4,894,704
SHUTTLE	4,896.4	802.6	113,980	804.2	819.2	**29,251.2**

Table 4: Average number of nodes and the resulting average memory consumptions.



Porovnanie priemernej spotreby pamäti náhodných lesov a rozhodovacích džunglí.

Hlavným cieľom práce je teda analyzovať a implementovať jednotlivé klasifikačné algoritmy na reálnych dátach, výsledky zhodnotiť a porovnať z hľadiska pamäťovej a časovej efektivity.

Literatúra

- Neuronal representations of distance in human auditory cortex https://pcl.ics.upjs.sk/wp-content/uploads/2014/10/Kopco-et-al-PNAS-2012-w-supp.pdf
- Decision Jungles: Compact and Rich Models for Classification https://www.microsoft.com/en-us/research/publication/decision-jungles-compact-and-rich-models-for-classification/

- Introduction to Machine Learning, Ethem Alpaydin
- Python Machine Learning, Sebastian Raschka
- Tobias Pohlen - Decision Jungles
- Klára Komprdová - Rozhodovací stromy a lesy