FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

META-UČENÍ
ENSEMBLE METHODS

Autor textu:
Ing. Petr Honzík, Ph.D.

Květen 2014

Komplexní inovace studijních programů a zvyšování kvality výuky na FEKT VUT v Brně
OP VK CZ.1.07/2.2.00/28.0193

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Obsah přednášky

1. Principy Meta-learningu
2. Bumping
3. Bagging
4. Stacking
5. Boosting
6. Shrnutí

Meta–learning = Ensemble methods

- Cíl – použít k predikci kombinaci více různých modelů
- Meta–learning (meta učení)
  - kombinování více modelů
  - metaznalost – znalost o znalosti
- Základní parametry (*co musíme určit?*)
  - jak použijeme (rozdělíme) trénovací data
  - jaké typy modelů kombinujeme
  - jak ve výsledku klasifikujeme

? Co je cílem meta-learningu?

Bumping – cesta k meta-learningu

- metoda využívající pouze **jeden model**, který je ale **vybraný z více modelů** naučených na jedněch trénovacích datech
- z trénovacích dat o *N* prvcích je vytvořeno *M* trénovacích množin náhodným výběrem s opakováním o *N* prvcích (metoda **bootstrap**) – použití pokud je **málo dat**
- **chyba** modelu je spočtena **ze všech trénovacích dat** a na jejím základě jsou vybrány nejlepší parametry modelu ***b*** z *M* vytvořených modelů

$$\hat{\mathbf{b}} = \arg\min_{b_j = b_1 \ldots b_M} \sum_{i=1}^{N} LF\left(y_i, \hat{f}\left(\mathbf{b}_j, x_i\right)\right)$$

- výsledky na nových testovacích datech dokládají, že tento způsob vede k výběru lepšího modelu než při výpočtu ze všech trénovacích dat (zejména robustní vůči uvíznutí v lokálních extrémech)

Bagging – idea

- Bagging = bootstrap aggregating
- Vytvořím z trénovacích dat *M* množin metodou bootstrap – tedy výběr s opakováním
- Z dat naučím *M* modelů **stejného typu** (např. rozhodovacích stromů)
- Při klasifikaci nechám *M* modelů hlasovat o konečné predikci, všechny modely jsou si **rovnocené** (mají stejnou váhu)
- vede ke zpřesnění (a stabilizaci) predikce, zhoršení pozorována jen ve výjimečných případech
- princip metody založen na teorii **bias-variance dekompozice**

Bagging – tvorba modelu

- máme *N* **trénovacích dat, vybereme** z nich **náhodně výběrem s opakováním *N* instancí** – jako v metodě bootsrap
  - pravděpodobnost výběru jednoho příkladu je *1/N*
  - při výběru *N* prvků je pravděpodobnost, že jeden konkrétní nebude ve výběru

$$\left(\frac{N-1}{N}\right)^N \underset{N\to\infty}{=} \frac{1}{e} = 0{,}368$$

  - teoreticky po výběru *N* prvků zbude *0,368.N* prvků na testování
- Tento postup zopakujeme *M*-krát, pokaždé jinou *N*-tici prvků, takže modely nebudeme trénovat na stejných datech
- **Učení modelu** pak vypadá tak, že vždy vybereme náhodně s opakováním *N* prvků, naučíme model a uložíme jeho nastavení. Máme tak na konci *M* různě nastavených modelů **stejného typu**.

Bagging - predikce

- máme novou instanci ***x***
- pro každý z *M* modelů urči třídu, do které má být prvek klasifikován
- vrať třídu, která byla **nejčastěji predikována**, dílčí modely jsou tedy rovnocené (*C* je počet tříd)

$$\hat{G}_{bagging}(x) = \arg\max_{i=1\ldots C} \sum_{j=1}^{M} \hat{f}_j(x)$$

- v případě regrese vrať **průměrnou** výstupní hodnotu

$$\hat{f}_{bagging}(x) = \frac{1}{M} \sum_{j=1}^{M} \hat{f}_j(x)$$

? Popiš princip a vlastnosti baggingu (v čem spočívá diverzifikace)

Bagging – vlastnosti, použití

- použití v případě **nedostatku dat**
- zvyšuje **stabilitu** a **přesnost (**snižuje **varianci** a riziko **přeučení)**
- typická aplikace na nestabilní modely (např. rozhodovací stromy, NN)
- předností RS je jejich interpretovatelnost, která se však použitím baggingu ztrácí
- nepoužívá se na lineární (velký bias, silná generalizace) a robustní (IBL) modely – nemá efekt
- existuje i váhová varianta MetaCost (pokud výstupy modelu mají pravděpodobnostní charakter – Bayes, NN)

Stacking – základní rysy

- učíme model, který kombinuje **libovolné typy modelů** nižší úrovně
- na vyšší úrovni učíme zpravidla jednoduchý (např. lineární) model
- je výhodné, když model nižší úrovně kromě predikce udává také míru důvěryhodnosti své predikce (logitový model, NN, Bayes, …)
- trénovacích data určena **výběrem typu Cross-Valiadation**, což je výpočetně náročné
- variantou je označení dílčích modelů **váhou w_i**, kterou je **násobena predikce každého modelu nižší úrovně**
- modely nižší úrovně, **úroveň 0,** jsou učeny vždy na základě stejné množiny trénovacích dat, jejich predikcí na testovacích datech získáme vstupní hodnoty do modelu **úrovně 1** (metalearner). Jeho struktura je jednoduchá (lineární, RS). Počet vstupů do meta-modelu odpovídá počtu modelů úrovně 0, nebo jeho násobku počtem tříd (když je výstupem nižšího modelů pravděpodobnost klasifikace do každé ze tříd)

V čem spočívá diverzifikace stackingu?

Stacking – varianta s výpočtem vah

- cílem je nastavit váhy w_i pro M modelů f_i nižší úrovně tak, aby byla minimalizována chyba LF od požadovaného výsledku
- $test$ představuje množinu testovacích dat. Zápis f^{-t} znamená model nastavený na základě trénovacích dat (bez t testovacích), x^{+t} testovací data, $|test|$ počet dělení pomocí Cross-Validation

$$w = \arg\min_{w} \sum_{t=1}^{|test|} LF\left(y^{+t}, \sum_{j=1}^{M} w_j \cdot f_j^{-t}\left(x^{+t}\right) \right)$$

- ideálně jsou váhy nastaveny tak, aby jejich součet byl roven 1

Boosting – idea

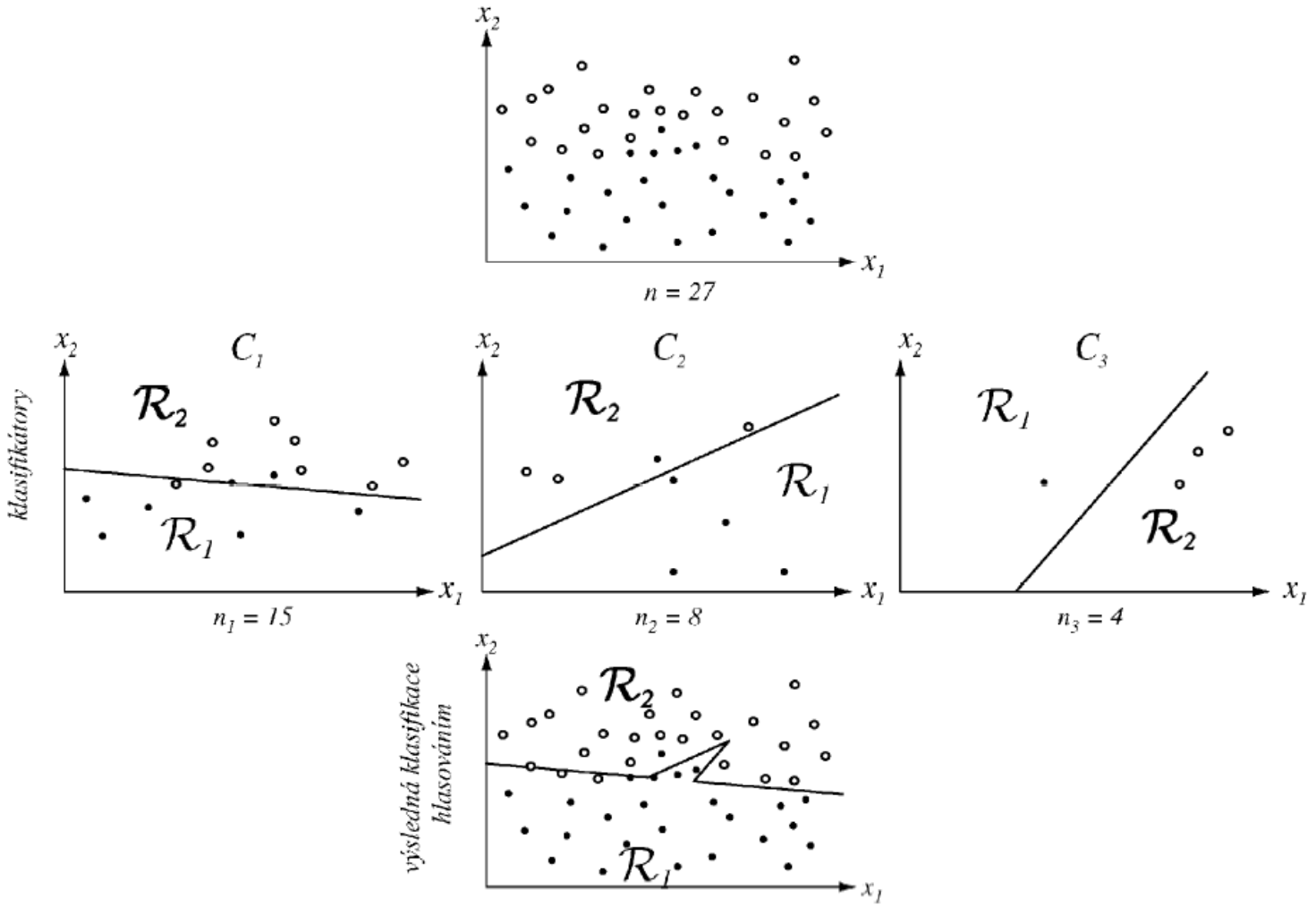
- nejpoužívanější metoda meta-learningu
- smyslem je vytvoření více **modelů stejného typu**, jejichž přesnost stačí lepší než náhodná, tzv. **weak learner** (více než 50% úspěšně klasifikovaných prvků)
- princip **diverzifikace** metody spočívá v tom, že jsou postupně vytvářeny modely více specializované na trénovací data, která model předešlý klasifikoval chybně
- různé algoritmy se liší způsobem, jakým zvyšují význam trénovacích dat, která nebyla v předešlých modelech rozpoznána
- předpoklad je, že více jednoduchých modelů pokryje přesněji prostor možných řešení

V čem spočívá diverzifikace boostingu?

Boosting – bez váhy (ostraňuje data)

- původní idea nepoužívá vah, jedinou diverzifikací je **cílený výběr trénovacích prvků** podle úspěšnosti klasifikace předešlých modelů
- na **první model** je použito např. *N/3* dat výběrem bez opakování, z nich nastaven první model f_1 s přesností lepší než 50%
- **další model** f_2 je nastaven z dat, která model f_1 klasifikuje chybně a z dalších dosud nepoužitých dat
- tak lze pokračovat do vyčerpání trénovací množiny
- **klasifikace** pak probíhá jako hlasování naučených modelů
- není jasně definováno, jakým způsobem jsou **data vybírána**, respektive není důkaz o tom, jaký způsob je nejlepší (výběry jsou však vždy bez opakování)

Boosting bez váhy – příklad

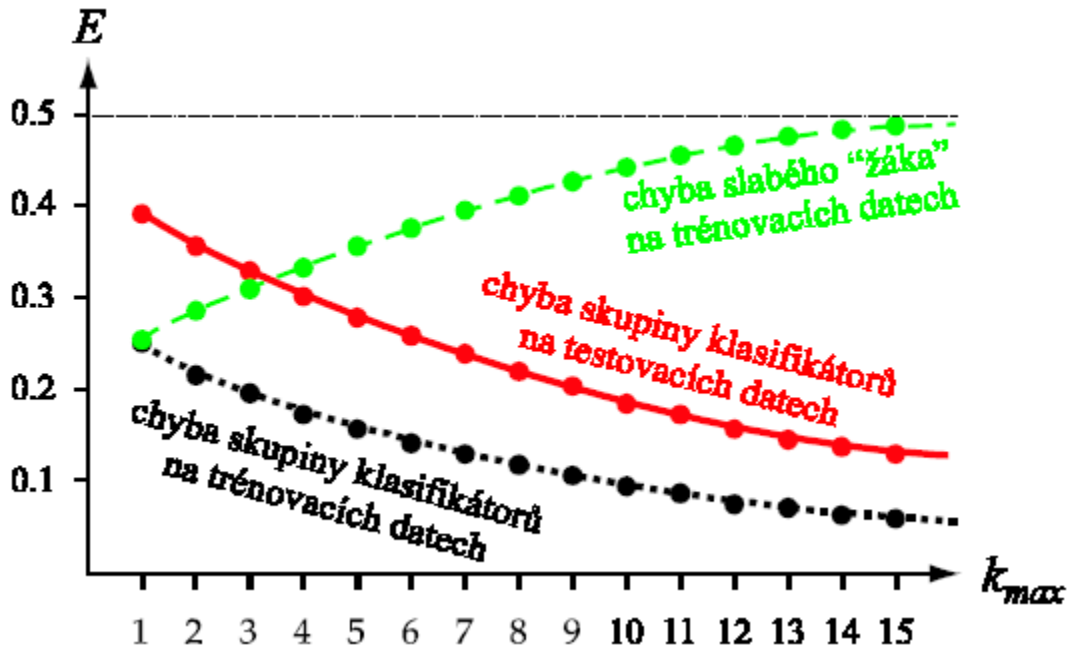


Boosting – s váhou (každý prvek má váhu)

- tato verze počítá s **váhami** jednotlivých záznamů
- model je vždy trénován na **všechna trénovací data**, avšak chybu klasifikace jednoho prvku ovlivňuje jeho váha (pokud ho předešlé modely vždy poznaly, má na chybu menší vliv)
- obecně se tento postup stále více specializuje na problémové prvky a vede k **větší chybě dílčího** modelu (nově vytvořeného)
- dále je každému vytvořenému modelu na základě jeho přesnosti přidělena **váha modelu** α, která určuje jeho vliv na konečnou klasifikaci
- **modifikací** je výběr nových trénovacích prvků s opakováním podle jejich pravděpodobnosti odvozené z velikosti váhy *w*

Boosting – celková chyba modelu

- s přibývajícím počtem modelů se jejich **dílčí chyba zvětšuje** (zelená), ale **celková chyba** jak na trénovacích (černá) tak na testovacích datech se **zmenšuje**



? Proč se u Boostingu s rostoucí iterací „dílčí trénovací chyba modelů“ zhoršuje

AdaBoost – princip

- AdaBoost – významný boostingový algoritmus
- dáno *N* záznamů ***x***,*y*
- váha každého prvku $W_1(i)=1/N$, kde $i=1..N$
- opakuj *k*-krát
  - nauč model C_k z dat s ohledem na váhy $W_k(i)$
  - z chyby E_k modelu C_k vypočti váhu modelu α_k
  - vytvoř W_{k+1} tak, že zvětšena váha chybně klasifikovaných, oslabena dobře klasifikovaných záznamů
- vytvořen klasifikátor tvořený *k* modely

? popište princip algoritmu AdaBoost

Boosting –základní algoritmus (binární klasifikace)

- **Algoritmus – učení**
  - nastav stejné váhy všem *N* prvkům, *k=1* (index iterace), opakuj:
    - nauč model s ohledem na váhy $\boldsymbol{W}_k$ prvků
    - vypočti chybu modelu E_k na všech trénovacích datech

$$E_k = \sum_{i=1}^{N} W_k(i) \cdot LF(y_i, C_k(x_i))$$

    - konec když E_k=0 nebo E_k>=0,5 (horší než náhodný)
    - přepočti $\boldsymbol{W}_k$ všech prvků a spočti α pro tento model, *k=k*+1

$$W_{k+1}(i) = W_k(i) \cdot \begin{cases} \frac{E_k}{1-E_k} & chybná\ klasifikace \\ 1 & správná klasifikace \end{cases} \qquad W_{k+1}(i) = \frac{W_{k+1}(i)}{\sum W_{k+1}(i)}$$

$$\alpha_k = -\log \frac{E_k}{1-E_k} = \log \frac{1-E_k}{E_k}$$

? Vyskytuje se v algoritmu boosting resubstituční chyba

AdaBoost – komentáře k algoritmu

- **Poznámka**
  - Kódování tříd –1 / +1
- **Odhad chyby modelu**
  - protože se počítá přes trénovací data, jedná se o chybu resubstituční (byť je model učen na datech váhovaných)
- **Váha dílčího klasifikátoru**
  - α_k je váha klasifikátoru C_k
- **Konečná klasifikace**

$$E_k = \sum_{i=1}^{N} W_k(i) \cdot LF(y_i, C_k(x_i))$$

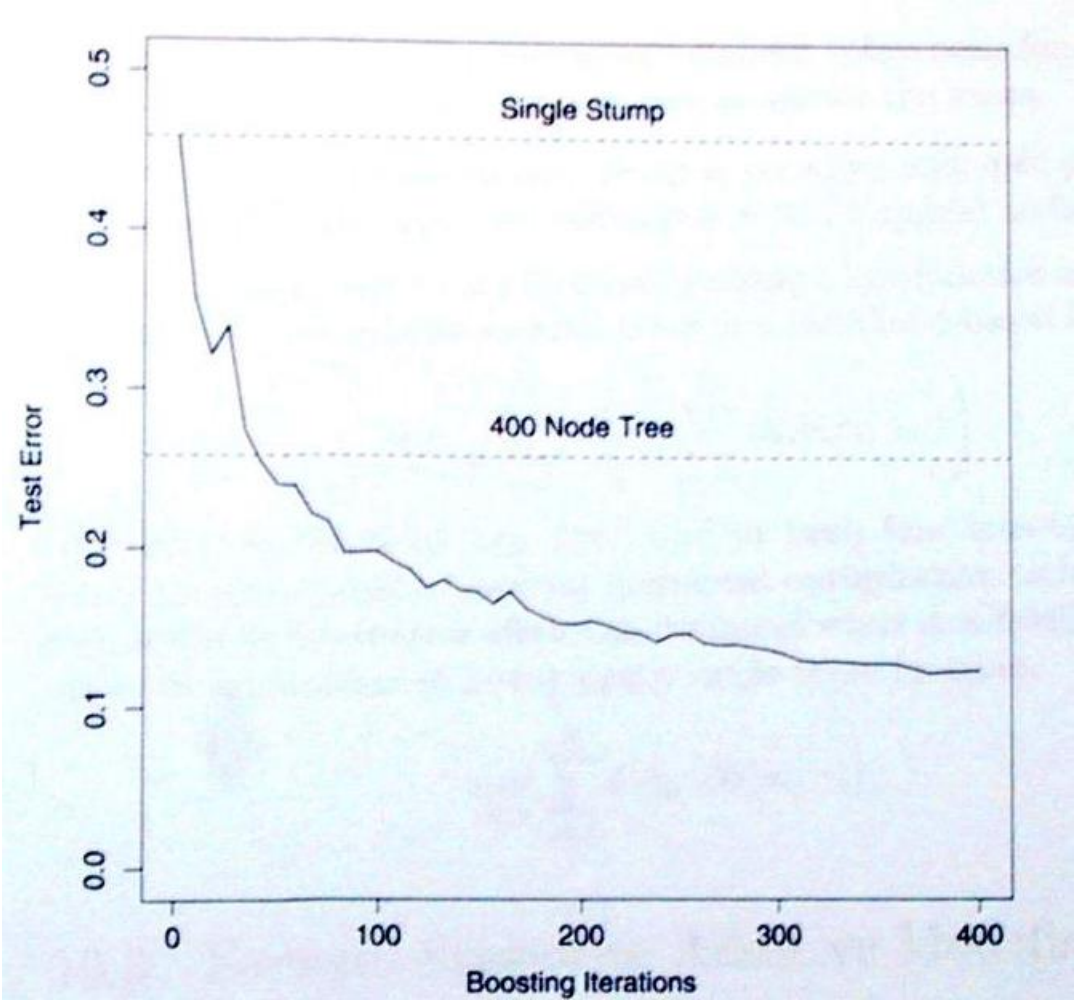
$$G(x) = sign \sum_{i=1}^{M} \alpha_i \cdot C_i(x)$$

AdaBoost - vlastnosti

- *AdaBoost* **redukuje trénovací chybu exponenciálně** v závislosti na rostoucím počtu k_{max} v hlasující skupině.
- pokud každý z klasifikátorů ve skupině dává lepší výsledky než náhodné, váhové **rozhodnutí** celé **skupiny** zaručuje pokles chyby na trénovacích datech
- velmi často dochází k podobnému poklesu i na testovacích datech
- pozor, algoritmus **je možné přeučit** (odvíjí se zejména od přílišné složitosti dílčích modelů)
- Occamovo ostří – identická chyba boosting modelů, druhý však z více modelů – empiricky zjištěno, že je na testovacích datech přesnější; není v rozporu, viz. hranice (meze) u separace optimální nadrovinou – respektive modely nejsou ve skutečnosti identické

AdaBoost – příklad

- binární klasifikace
- 10 vstupních veličin
- všechny mají normální rozložení
- 2.000 trénovacích
- 10.000 testovacích
- sledujeme chybu na testovacích datech (test error)



Hlavní charakteristiky 1/2

- **Bagging**
  - trénovacích dat *N*, náhodným výběrem s opakováním *M* trénovacích množin, použití když máme málo dat a nestabilní prediktor (RS, NN)
  - *M* modelů stejného typu naučené z *M* trénovacích výběrů
  - diverzifikace různými trénovacími množinami
  - jednotlivé modely jsou rovnocené, rozhoduje nejčastější výsledek (klasifikace) nebo průměr (regrese)
- **Stacking**
  - použití dělení dat typu Cross-Validation
  - modely různého typu naučené vždy ze stejných trénovacích dat
  - diverzifikace různými algoritmy
  - každý model má svou váhu (míru důvěry ve svou predikci)

Hlavní charakteristiky 2/2

- **Boosting**
  - trénovací data váhována, posilování chybně predikovaných záznamů
  - použití M modelů stejného typu
  - diverzifikaci tvoří postupně se měnící váhy trénovacích dat v závislosti na tom, jak úspěšně jsou klasifikovány jednotlivými modely
  - klasifikace probíhá na základě hlasování jednotlivých modelů; je brána v potaz váha každého modelu