Katedra Geomatiky

Strojové učení v dálkovém průzkumu Země

155YUSU

Lekce 3: Fundamentální algoritmy strojového učení
{lineární a logistická regrese, rozhodovací stromy, random forest, metoda podpůrných vektorů SVM, KNN}

Přednášející: Lukáš Brodský lukas.brodsky@natur.cuni.cz , Martin Landa martin.landa@fsv.cvut.cz a Ondřej Pešek ondrej.pesek@fsv.cvut.cz

Lineární regrese

Lineární regrese

- Model učící se lineární kombinaci daných hodnot

Problém

- Máme soustavu označkovaných (labelled) hodnot

$$\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$$

kde $\mathbf{x}_i$ je D-rozměrný vektor daných hodnot
a y_i je cílová hodnota;

Problém

- Úkolem je vytvořit model **f(x)** jakožto lineární kombinaci daných hodnot x:

$$f_{w,b}(\mathbf{x}) = \mathbf{w}\mathbf{x} + b,$$

- kde **w** je D-rozměrný vektor vah (parametrů) a b je reálné číslo.

Povšimněte si, že předpis našeho lineárního modelu je velmi podobný předpisu SVM… který ale ukážeme až později

Praktické ospravedlnění lineárního modelu:

- Je jednoduchý.
- Jeho přeučení je výjimečné.

Problém

- **Úkol**: minimalizuj chyby modelu, nalezni optimální hodnoty (**w***, b*), s kterými se budeme nejvíce blížit cílovým hodnotám.

- Podmínka: Nadrovina regrese leží tak blízko trénovací množině, jak je možno.

Řešení

- Optimalizační postup (optimization routine): Nalezni optimální hodnoty w* a b* tak, aby se minimalizovala účelová funkce:

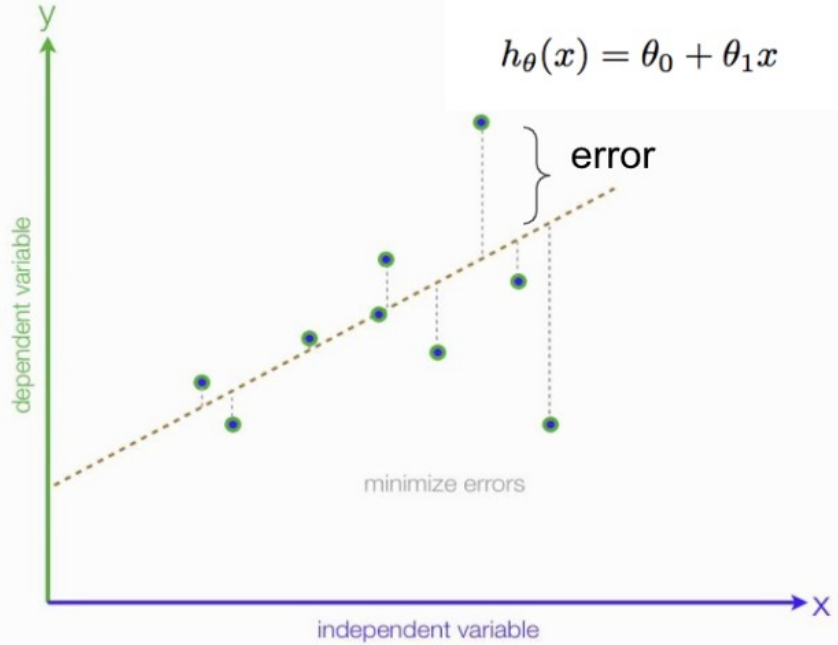
$$\frac{1}{N} \sum_{i=1 \ldots N} (f_{\mathbf{w},b}(\mathbf{x}_i) - y_i)^2.$$

kde $(f_{w,b}(x_i) – y_i)^2$ je ztrátová funkce *(metrika postihu za špatnou klasifikaci)*;

Poznámky

- Všechny modely strojového učení využívají ztrátových funkcí
- V lineární regresi se jako ztrátová funkce používá průměrná ztráta, také nazývána empirický risk (empirical risk, ER)

Lineární regrese graficky



$$h_\theta(x) = \theta_0 + \theta_1 x$$

Hypothesis:

$$h_\theta(x) = \theta_0 + \theta_1 x$$

Parameters:

$$\theta_0, \theta_1$$

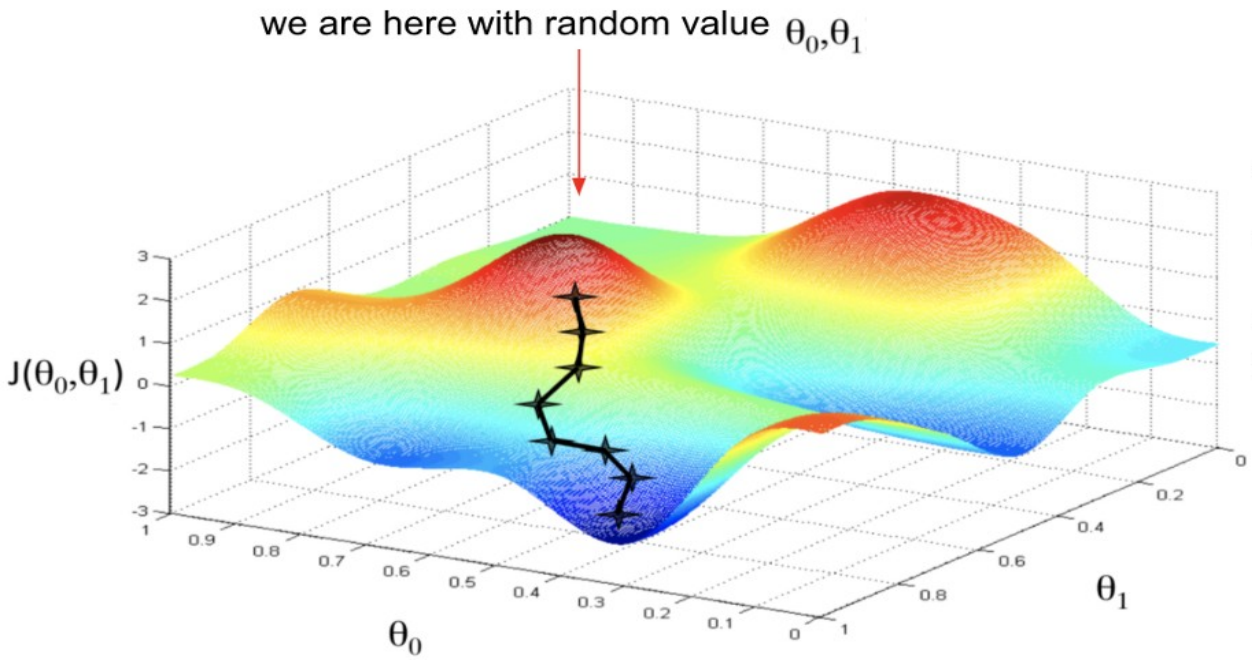
Cost Function:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

Goal:

$$\underset{\theta_0, \theta_1}{minimize} J(\theta_0, \theta_1)$$

Optimalizační postup: SGD (stochasticky klesající gradient)



- Start with some θ_0,θ_1
- Keep changing θ_0,θ_1 to reduce $J(\theta_0,\theta_1)$

until we hopefully end up at a minimum

Anatomie algoritmu

1/ Mějme model lineární regrese;

2/ definujme ztrátovou funkci (postih);

3/ zvolme optimalizační kritérium;

4/ zvolme optimalizační postup;

Logistická regrese

Logistická regrese

- Není to regrese, nýbrž klasifikační učící se algoritmus.

Problém

- Opět vytváříme model y_i jakožto lineární funkci

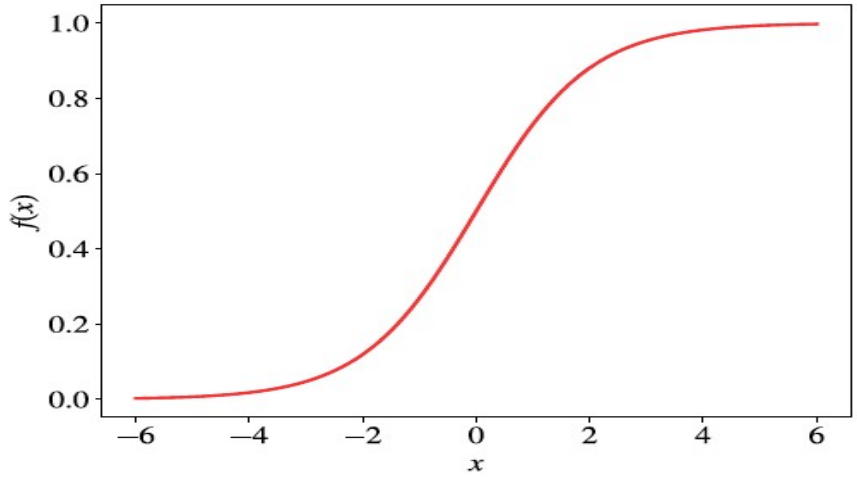
$$y_i = \boldsymbol{w}x_i + b$$

kde jednotlivé proměnné nabírají hodnot hodnot od -∞ do +∞, ale y_i nabývá pouze dvou hodnot.

=> potřeba jednoduché spojité funkce k získání hodnot pravděpodobnosti mezi **(0, 1)**

Logistická funkce (sigmoid)

$$f(x) = \frac{1}{1 + e^{-x}},$$



Model logistické regrese

$$f_{\mathbf{w},b}(\mathbf{x}) \stackrel{\text{def}}{=} \frac{1}{1+e^{-(\mathbf{w}\mathbf{x}+b)}}$$

- Povšimněte si povědomého *wx + b*.

Model logistické regrese

$$f_{\mathbf{w},b}(\mathbf{x}) \stackrel{\text{def}}{=} \frac{1}{1+e^{-(\mathbf{w}\mathbf{x}+b)}}$$

- **Úkol**: optimalizujte hodnoty ***w*** a *b*;
- Výstup *f(x)* jsou pravděpodobnosti, že y_i je pozitivní třídou;

f(x) > 0.5 -> *y* je pozitivní, jinak negativní

Řešení

- Snažíme se maximalizovat věrohodnost (likelihood) modelu vůči trénovacím datům

Optimalizační kritérium

- Namísto minimalizace průměrné ztráty maximalizujeme věrohodnost modelu.

$$L_{\mathbf{w},b} \overset{\text{def}}{=} \prod_{i=1\ldots N} f_{\mathbf{w},b}(\mathbf{x}_i)^{y_i} (1 - f_{\mathbf{w},b}(\mathbf{x}_i))^{(1-y_i)}$$

pouze $f_{w,b}(x)$ když $y_i=1$ protože $(1 - y_i) = 0$

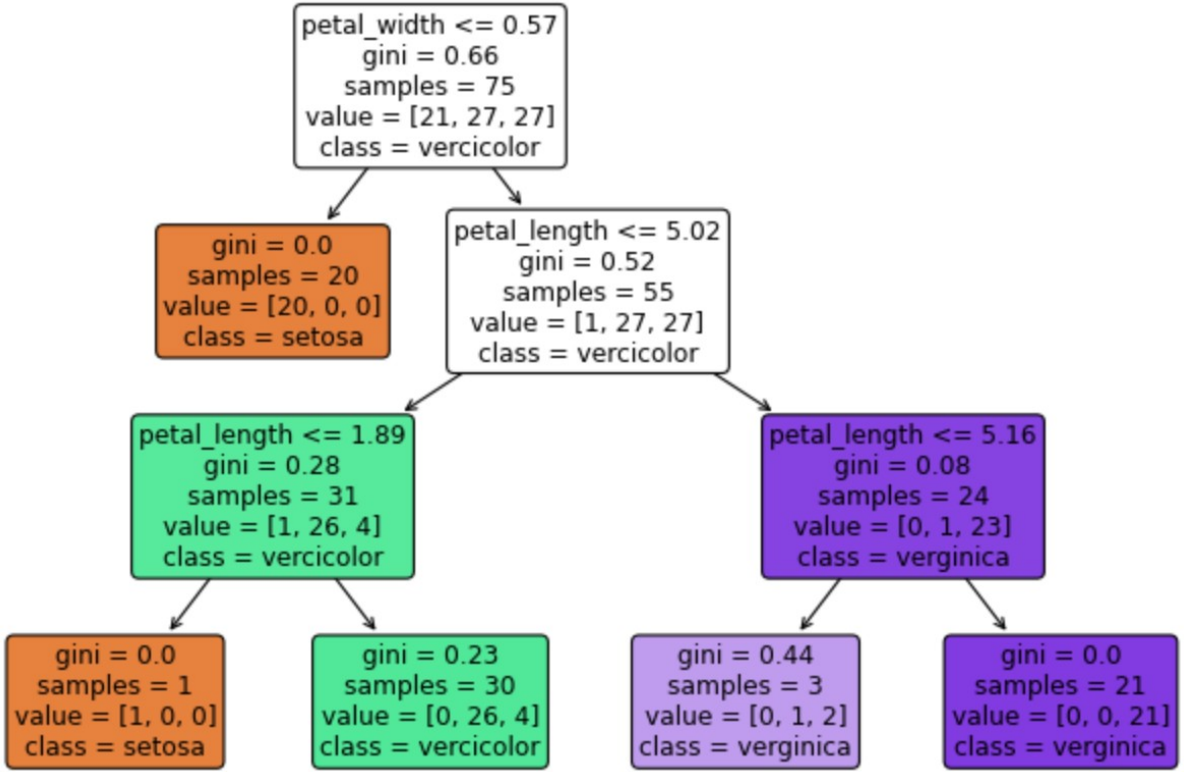
a $(1 - f_{w,b}(x))$ když $y_i = 0$ protože $f(x)^y = 1$;

Optimalizační kritérium

- Prakticky: z důvodu vyhnutí se přetečení exponenciální funkce je vhodné využívat log-likelihood

$$LogL_{\mathbf{w},b} \stackrel{\text{def}}{=} \ln(L(_{\mathbf{w},b}(\mathbf{x}))$$

Rozhodovací stromy


petal_width <= 0.57
gini = 0.66
samples = 75
value = [21, 27, 27]
class = vercicolor
gini = 0.0
samples = 20
value = [20, 0, 0]
class = setosa
petal_length <= 5.02
gini = 0.52
samples = 55
value = [1, 27, 27]
class = vercicolor
petal_length <= 1.89
gini = 0.28
samples = 31
value = [1, 26, 4]
class = vercicolor
petal_length <= 5.16
gini = 0.08
samples = 24
value = [0, 1, 23]
class = verginica
gini = 0.0
samples = 1
value = [1, 0, 0]
class = setosa
gini = 0.23
samples = 30
value = [0, 26, 4]
class = vercicolor
gini = 0.44
samples = 3
value = [0, 1, 2]
class = verginica
gini = 0.0
samples = 21
value = [0, 0, 21]
class = verginica

Rozhodovací stromy

- Princip: acyklický rozhodovací graf
- Pravidlo: **prvek$_j$ < práh**
  - > následuj levou větev; jinak následuj pravou
- Vnitřní uzly: otestuj atributy;
- Vnější uzly (listy): rozhodni o třídě

Problém

- Mějme množinu (set) označkovaných příkladů:

  set {0, 1};

- Vytvořme rozhodovací strom s listy odpovídajícími zájmovým třídám

  => hledej optimální pravidla, která rozdělí nejméně podobná data, dokud nedosáhneš požadovaného stupně podobnosti

Řešení

- S .. množina označkovaných příkladů

- Úkol: Vytvoř rozhodovací strom, který vypočte *y* na základě *x*
- Iterativně rozděluj momentální data do dvou větví

Řešení

- 1. iniciace: konstantní model
- Odhad *y* bude totožný pro všechna *x*

Řešení

- 2. rekurzivně procházej všechny prvky (features) *j = 1,…,J* a všechny prahy *t = 1..T* takové, že dělí data *S* do dvou podmnožin: *S*- and *S*+
  - Pro každý pár *(j, t)* ohodnoťme, jak dobře data rozděluje
- 3. **vyberme nejlepší** *(j, t)*;

Řešení

- 4. Algoritmus učiní z uzlu koncový v těchto situacích:
  - Všechny příklady v uzlu byly klasifikovány;
  - Strom nabyl maximální hloubky *d*
  - Nelze nalézt parametry, podle kterých data dělit
  - Rozdělení snižuje entropii o méně než *epsilon*

Řešení

Pravidla dělení? = kvalita dělení

- Information Entropy gain nebo Gini impurity

Řešení

- **Entropie** ~ kolik variance data obsahují.
  - podmnožina jedné třídy dat bude mít nulovou entropii
  - podmnožina z rozdílných tříd bude mít vyšší entropii

$$E = -\sum_{i}^{C} p_i \log_2 p_i$$

kde p_i je pravděpodobnost, že z množiny dat náhodně vybereme třídu *i* (tj. zastoupení třídy *i* v datasetu).

Entropie - příklad

- Uvažujme dataset s 6 prvky: 1 třídy B, 2 třídy Gs, 3 třídy Rs;

$$E = -(p_b \log_2 p_b + p_g \log_2 p_g + p_r \log_2 p_r)$$

$$E = -(\frac{1}{6} \log_2(\frac{1}{6}) + \frac{2}{6} \log_2(\frac{2}{6}) + \frac{3}{6} \log_2(\frac{3}{6}))$$

$$= \boxed{1.46}$$

Entropie - příklad

- Dataset s 3 prvky třídy B

$$E = -(1 \log_2 1) = \boxed{0}$$

Řešení

Jak můžeme kvantifikovat kvalitu rozdělení?

- 1. spočteme entropii před rozdělením a po něm
- 2. určíme kvalitu rozdělení pomocí vážené entropie každé větve (na základě počtu prvků)

Information gain = **kolik entropie odstraníme**.

Gain = 1 - entropy

Řešení

- **Gini impurity**: pravděpodobnost nesprávné klasifikace náhodně vybraného prvku v datasetu, byla-li by jeho třída náhodně určena na základě rozdělení tříd v datasetu.

$$G = \sum_{i=1}^{C} p(i) * (1 - p(i))$$

kde *C* je počet tříd a *p(i)* je pravděpodobnost náhodného výběru prvku třídy *i*.

- Cíl učení rozhodovacích stromů: **maximalizovat Gini gain** (minimalizovat Gini impurity).

Problémy

- Rozdělení se děje lokálně při každé iteraci -> algoritmus nemusí dosáhnout optimálního řešení

- Prořez odspoda vzhůru řeší přetrénování
  - Odstraňují se větve buď nejméně důvěryhodné, nebo s nejméně vzorky

Parametry

- **max_depth**: maximální hloubka stromu
- **min_samples_split**: nejmenší počet vzorků nutný k rozdělení v uzlu – vždy je však zapotřebí alespoň dvou vzorků
- **min_samples_leaf**: nejmenší počet vzorků nutný k vytvoření listu – koncového uzlu. Tento parametr pomahá ve vyhlazení modelu