Klasifikace a strojové učení

Ing. Lukáš Bureš
Katedra kybernetiky

Plzeň
14. srpna 2014

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Předmluva

Tento studijní materiál vznikl na Západočeské univerzitě v Plzni, Fakultě aplikovaných věd, Katedře kybernetiky jako pomocný podklad pro předměty Zpracování digitalizovaného obrazu (KKY/ZDO) a Metody počítačového vidění (KKY/MPV) v roce 2014.

CV je nejlepší, moderní atd.

Machine learning .

Kapitola 2

Úvod

2.1 Co je strojové učení?

Pojem strojové učení nemá dodnes pevně danou definici, můžeme si ukázat několik pokusů o jeho definování

- Arthur Samuel (1959). Strojové učení: *Obor, který dává počítačům schopnost učit se bez toho, aby byly přímo naprogramovány.*
- Tom Mitchell (1998): Dobře definovaný problém strojového učení: *Předpokladem je, že se počítačový program učí ze zkušeností* $\boldsymbol{E}$ *s respektováním úlohy* $\boldsymbol{T}$ *a s měřičem výkonu* $\boldsymbol{P}$*, pokud se jeho výkon na úloze* $\boldsymbol{T}$ *měřený pomocí* $\boldsymbol{P}$ *zvyšuje se zkušeností* $\boldsymbol{E}$*.*

Pokusme se jednotlivé části rozebrat na příkladu: Předpokládejme, že náš emailový klient sleduje, které email označíme nebo neoznačíme jako spam. Na základě tohoto rozhodnutí se náš emailový klient učí lépe rozpoznávat co je a co není spam. Nyní si můžeme položit otázku co z následujících tvrzení je úloha **T**?

1. Klasifikace emailů jako spam nebo vyžádaný email.
2. Sledování značek od uživatele, který email označil jako spam nebo vyžádaný email.
3. Počet (nebo poměr) emailů, které byly správně klasifikovány jako spam nebo vyžádaný email.
4. Nic z výše uvedeného - není to problém strojového učení.

Pokud si rozebereme jednotlivé body výše, tak bod jedna je definice úlohy **T**, dále bod dva je zkušenost **E** a třetí bod je náš měřič výkonu **P**.

Existuje několik rozdílných typů učících se algoritmů. Hlavní dva typy se nazývají

- učení s učitelem a
- učení bez učitele.

2.2 Učení s učitelem

Učení s učitelem je metoda strojového učení pro učení funkce z trénovacích dat. Trénovací data sestávají ze dvojic vstupních objektů (typicky vektorů příznaků) a požadovaného výstupu. Výstup funkce může být spojitá hodnota (při regresi) anebo může předpovídat označení třídy vstupního objektu (při klasifikaci). Úloha algoritmu učení je předpovídat výstupní hodnotu funkce pro každý platný vstupní objekt poté, co zpracuje trénovací příklady (tj. dvojice vstup a požadovaný výstup). Aby to dokázal, musí algoritmus zobecnit prezentovaná data na nové situace (vstupy) „smysluplným“ způsobem. Analogická úloha v lidské a zvířecí psychologii se často nazývá učení konceptů.

Přetrénování (anglicky *overfitting*) je stav, kdy je systém příliš přizpůsoben množině trénovacích dat, ale nemá schopnost generalizace a selhává na testovací (validační) množině dat. To se může stát například při malém rozsahu trénovací množiny nebo pokud je systém příliš komplexní (například příliš mnoho skrytých neuronů v neuronové síti). Řešením je zvětšení trénovací množiny, snížení složitosti systému nebo různé techniky regularizace, zavedení omezení na parametry systému, které v důsledku snižuje složitost popisu naučené funkce, nebo předčasné ukončení (průběžné testování na validační množině a konec učení ve chvíli, kdy se chyba na této množině dostane do svého minima).

Při učení se používají trénovací data (nebo trénovací množina), testovací data a často validační data.

Příklady algoritmů: rozhodovací stromy, AdaBoost, náhodné rozhodovací lesy, metoda nejbližšího souseda, metoda K-nejbližších sousedů, lineární regrese, Bayesův klasifikátor, neuronové sítě, binární regrese, support vector machine (SVM), atd.

2.3 Učení bez učitele

Příklady algoritmů: BIRCH, hierarchické algoritmy, divizní algoritmy, K-means (MacQueen algoritmus), expectation-maximization (EM) algoritmus, atd.

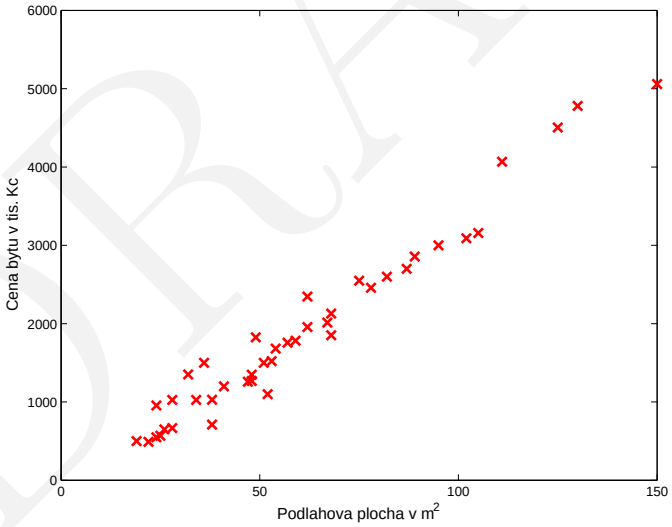
Kapitola 3

Lineární regrese

K cemu slouzi regrese

3.1 Motivace

Jako motivaci můžeme použít příklad, kdy si budete chtít zakoupit byt v Plzni, proto by bylo dobré znát, kolik zaplatíte za jakou podlahovou plochu, v ideálním případě pro všechny velikosti bytů. Nebo-li se snažíte predikovat cenu bytu v závislosti na podlahové ploše. Na Obr. 3.1 lze vidět vývoj ceny bytu v závislosti na podlahové ploše.



Obrázek 3.1: Vývoj ceny bytu v závislosti na podlahové ploše.

3.2 Učení s učitelem

Učení s učitelem (supervised learning) dává „správnou odpověď“ pro každý příklad z množiny dat.

Problém regrese: Na základě reálného (spojitého) vstupu predikovat reálný (spojitý) výstup.

Problém klasifikace: Na základě diskrétního vstupu predikovat diskrétní výstup.

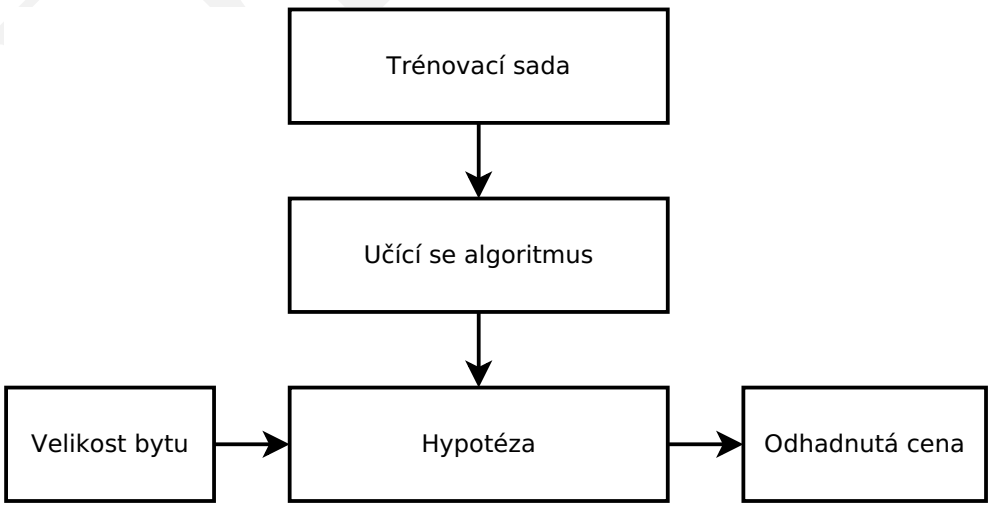
Více formálně v případě učení s učitelem potřebujeme množinu dat nazývanou trénovací množina. V našem případě predikce ceny bytu máme trénovací množinu cen bytů a k nim odpovídající podlahovou plochu. Naším úkolem je naučit se z těchto dat jak predikovat ceny bytů.

Nyní zavedeme notaci nechť m je velikost trénovací sady, x jsou „vstupní“ proměnné neboli příznaky a y jsou „výstupní“ proměnné neboli příznaky. Jeden trénovací vzorek budeme označovat jako $(\boldsymbol{x}, y)$ a odpovídá jednomu řádku v tabulce 3.1. Dále $\left(x^{(i)}, y^{(i)}\right)$ označuje i-tý trénovací vzorek.

Velikost bytu v m^2 (x)	Cena bytu v tis. Kč (y)
19	500
22	490
25	568
...	...

Tabulka 3.1: Příklady trénovacích vzorků.

Trénovací množina, v našem případě ceny bytů, je potřebná pro natrénování učícího se algoritmu, jehož výstupem je funkce (hypotézy), která se většinou označuje h. Účelem funkce (hypotézy) h je ze vstupu x (velikost bytu) odhadnout výstup y (cena bytu). Tedy h je funkce, která mapuje x na y. Schéma je naznačeno na Obr. 3.2.



Obrázek 3.2: Schéma učení s učitelem.

Když navrhujeme učící se algoritmy, další věc kterou potřebujeme rozhodnout je jak budeme reprezentovat hypotézu h. V našem motivačním příkladě si vystačíme s hypotézou v následujícím tvaru lineární funkce:

$$h_{\boldsymbol{\Theta}}(x) = \vartheta_0 + \vartheta_1 x, \tag{3.1}$$

zkráceně $h(x)$. Jedná se tedy o lineární regresi s jednou proměnnou. V případě komplexnějšího učícího se algoritmu je nutné použít složitější nelineární model.

Poznámka

ϑ je malá théta a Θ je velká théta.

3.3 Ztrátová funkce

Pro nejlepší napasování našeho lineárního modelu na trénovací data budeme potřebovat určit parametr ϑ_0 a ϑ_1 v naší hypotéze:

$$h_{\mathbf{\Theta}}(x) = \vartheta_0 + \vartheta_1 x, \tag{3.2}$$

Snažíme se zvolit parametry ϑ_0 a ϑ_1 tak, aby hypotéza $h_{\mathbf{\Theta}}(x)$ byla co nejblíže k y pro naše trénovací vzorky (x, y). Jinými slovy snažíme se minimalizovat kvadratickou chybu:

$$\min_{\vartheta_0,\vartheta_1} \frac{1}{2m} \sum_{i=1}^{m} \left(h_{\mathbf{\Theta}}\left(x^{(i)}\right) - y^{(i)} \right)^2, \tag{3.3}$$

kde

$$h_{\mathbf{\Theta}}\left(x^{(i)}\right) = \vartheta_0 + \vartheta_1 x^{(i)}. \tag{3.4}$$

Přesněji označíme ztrátovou funkci:

$$J(\vartheta_0, \vartheta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_{\mathbf{\Theta}}\left(x^{(i)}\right) - y^{(i)} \right)^2 \tag{3.5}$$

a minimalizujeme

$$\min_{\vartheta_0,\vartheta_1} J(\vartheta_0, \vartheta_1). \tag{3.6}$$

Tato ztrátová funkce J je také občas nazývána kvadratická chyba nebo ztrátová funkce kvadratické chyby.

Ztrátová funkce ve formě kvadratické chyby je nejčastěji používanou ztrátovou funkcí při řešení regresních problémů.

3.4 Gradientní algoritmus

V kapitole 3.3 jsme definovali ztrátovou funkci J. V této kapitole bude popsán gradientní algoritmus pro minimalizaci ztrátové funkce J. Gradientní algoritmus není omezen jen a pouze na lineární regresi nalézá uplatnění v optimalizaci a v dalších oblastech strojového učení.

Máme ztrátovou funkci $J(\vartheta_0, \vartheta_1)$ a chtěli bychom jí minimalizovat. Proto zvolíme náhodně ϑ_0 a ϑ_1, dále v každé iteraci měníme hodnoty ϑ_0 a ϑ_1, tak abychom redukovali $J(\vartheta_0, \vartheta_1)$ dokud, v nejlepším případě, nedokonvergujeme do globálního minima.

Obecně lze gradientní algoritmus použít na jakoukoliv ztrátovou funkci:

$$J(\vartheta_0, \cdots, \vartheta_n) \tag{3.7}$$

a snažíme se minimalizovat:

$$\min_{\vartheta_0,\ldots,\vartheta_n} J(\vartheta_0, \ldots, \vartheta_n) \tag{3.8}$$

Definice gradientního algoritmu: opakuj dokud není dosaženo konvergence

$$\vartheta_j = \vartheta_j - \alpha \frac{\partial}{\partial \vartheta_j} J(\vartheta_0, \vartheta_1), \tag{3.9}$$

pro $j = \{0, 1\}$, pro náš příklad (lineární regrese jedné proměnné) v sekci 3.1. V rovnici 3.9 proměnná α určuje, jak velké kroky bude gradientní algoritmus dělat při hledání minima a nazývá se rychlost učení. V případě velké hodnoty α bude algoritmus postupovat velkými kroky, rychle, ale není zajištěna konvergence, jelikož se může stát, že algoritmus bude kroužit kolem minima, ale nedosáhne ho, nebo bude divergovat. V případě malé hodnoty α bude algoritmus velice pomalý a nemusí dosáhnout minima pro stanovený počet iterací. Gradientní algoritmus může konvergovat do lokálního minima v případě, že je rychlost učení α konstantní. Proto se v reálných aplikacích volí nejdříve větší α, které se s rostoucí hodnotou iterací nebo času zmenšuje.

Poznámka

Pro implementaci: je nutné aktualizovat všechny Θ současně

$$temp_0 = \vartheta_0 - \alpha \frac{\partial}{\partial \vartheta_0} J(\vartheta_0, \vartheta_1), \tag{3.10}$$

$$temp_1 = \vartheta_1 - \alpha \frac{\partial}{\partial \vartheta_1} J(\vartheta_0, \vartheta_1), \tag{3.11}$$

$$\vartheta_0 = temp_0, \tag{3.12}$$

$$\vartheta_1 = temp_1. \tag{3.13}$$

Nyní budeme aplikovat gradientní algoritmus na náš problém lineární regrese

$$\frac{\partial}{\partial \vartheta_j} J(\vartheta_0, \vartheta_1) = \frac{\partial}{\partial \vartheta_j} \cdot \frac{1}{2m} \sum_{i=1}^{m} \left(h_{\mathbf{\Theta}}\left(x^{(i)}\right) - y^{(i)} \right)^2, \tag{3.14}$$

po dosazení z rovnice 3.4 dostáváme

$$\frac{\partial}{\partial \vartheta_j} J\left(\vartheta_0, \vartheta_1\right) = \frac{\partial}{\partial \vartheta_j} \cdot \frac{1}{2m} \sum_{i=1}^{m} \left(\vartheta_0 + \vartheta_1 x^{(i)} - y^{(i)}\right)^2, \tag{3.15}$$

po zderivování podle jednotlivých parametrů Θ získáme

$$\frac{\partial}{\partial \vartheta_0} \quad J\left(\vartheta_0, \vartheta_1\right) \quad = \frac{1}{m} \sum_{i=1}^{m} \left(h_{\mathbf{\Theta}}\left(x^{(i)}\right) - y^{(i)}\right), \tag{3.16}$$

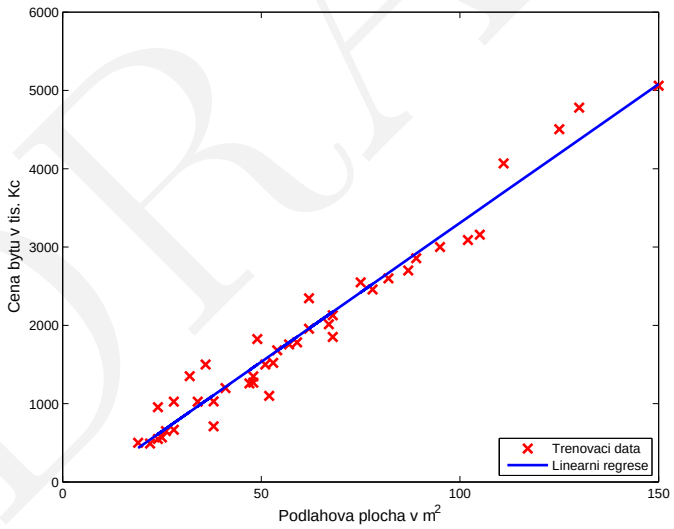
$$\frac{\partial}{\partial \vartheta_1} \quad J\left(\vartheta_0, \vartheta_1\right) \quad = \frac{1}{m} \sum_{i=1}^{m} \left(h_{\mathbf{\Theta}}\left(x^{(i)}\right) - y^{(i)}\right) x^{(i)}. \tag{3.17}$$

Výsledný konkrétní tvar rovnice 3.9 pro ϑ_0 a ϑ_1

$$\vartheta_0 \quad = \vartheta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_{\mathbf{\Theta}}\left(x^{(i)}\right) - y^{(i)}\right), \tag{3.18}$$

$$\vartheta_1 \quad = \vartheta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_{\mathbf{\Theta}}\left(x^{(i)}\right) - y^{(i)}\right) x^{(i)}. \tag{3.19}$$

Gradientnímu algoritmu popsanému výše se také někdy říká dávkový gradientní algoritmus, protože využívá všechna trénovací data, při součtu přes všech m vzorků, v každém kroku.



Obrázek 3.3: Výsledná lineární regrese našeho úvodního příkladu po výpočtu gradientním algoritmem.

3.5 Vícerozměrná lineární regrese

V sekci 3.2 byla představena lineární regrese o jedné proměnné na příkladě predikce ceny bytu (výstup y) v závislosti na velikosti podlahové plochy m^2 (vstup x).

V případě vícerozměrné lineární regrese si můžeme představit, že nemáme pouze velikost podlahové plochy bytu, ale výsledná cena závisí i dalších parametrech bytu, viz tabulka 3.2.

Velikost bytu x_1	Počet místností x_2	Počet poschodí x_3	Stáří bytu x_4	Cena bytu y
19	5	1	45	500
22	3	2	40	490
25	2	1	30	568
...	...	...	...	...

Tabulka 3.2: Příklady vícerozměrných trénovacích vzorků.

Proměnnou n budeme označovat počet příznaků (v našem případě v tabulce 3.2 $x_{1,\dots,4}$, proto $n = 4$.) Dále budeme stále jako u jednorozměrného případu označovat vstupní příznaky (všechny = sloupcový vektor) $\boldsymbol{x}^{(i)}$ pro i-tý trénovací vzor. A hodnota jednotlivého j-tého příznaku i-tého trénovacího vzoru bude označována $\boldsymbol{x}_j^{(i)}$.

V případě jednorozměrné lineární regrese byla naše hypotéza h_Θ reprezentována

$$h_{\boldsymbol{\Theta}}(x) = \vartheta_0 + \vartheta_1 x, \tag{3.20}$$

v našem vícerozměrném případě nabývá tvaru

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \vartheta_0 + \vartheta_1 x_1 + \vartheta_2 x_2 + \vartheta_3 x_3 + \vartheta_4 x_4. \tag{3.21}$$

V případě 3.20 i 3.21 je možné si všimnout, že $x_0 = 1$ čehož dále využijeme, přesněji $\boldsymbol{x}_0^{(i)} = 1$ z důvodu dodržení konvence. Proto $\boldsymbol{x} \in \mathrm{R}^{n+1}$ a to samé platí pro parametry $\boldsymbol{\Theta} \in \mathrm{R}^{n+1}$.

Poznámka

V našem příkladě odpovídají

$$\boldsymbol{x} = [x_0,\ x_1,\ x_2,\ x_3,\ x_4]^\top \tag{3.22}$$

a

$$\boldsymbol{\Theta} = [\vartheta_0,\ \vartheta_1,\ \vartheta_2,\ \vartheta_3,\ \vartheta_4]^\top. \tag{3.23}$$

Obecněji lze vícerozměrnou hypotézu zapsat jako

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \vartheta_0 + \vartheta_1 x_1 + \vartheta_2 x_2 + \ldots + \vartheta_n x_n, \tag{3.24}$$

neboli jednoduše jako součin vektorů (vnitřní produkt)

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \boldsymbol{\Theta}^\top \boldsymbol{x}. \tag{3.25}$$

3.6 Gradientní algoritmus - vícerozměrná verze

Budeme potřebovat hypotézu

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \boldsymbol{\Theta}^{\top}\boldsymbol{x} = \vartheta_0 x_0 + \vartheta_1 x_1 + \vartheta_2 x_2 + \ldots + \vartheta_n x_n, \tag{3.26}$$

kde $x_0 = 1$, dále parametry $\boldsymbol{\Theta} = [\vartheta_0,\ \vartheta_1,\ \vartheta_2,\ \ldots,\ \vartheta_4]^{\top}$ a vícerozměrnou ztrátovou funkci

$$J(\boldsymbol{\Theta}) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2. \tag{3.27}$$

Gradientní algoritmus pak bude mít tvar: opakuj

$$\vartheta_j = \vartheta_j - \alpha\frac{\partial}{\partial\vartheta_j}J(\boldsymbol{\Theta}), \tag{3.28}$$

současně aktualizovat parametry pro všechny $j = 0, \ldots, n$.

Vztah z rovnice 3.28 lze zapsat jako

$$\frac{\partial}{\partial\vartheta_j}J(\boldsymbol{\Theta}) = \frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)\boldsymbol{x}_j^{(i)} \tag{3.29}$$

a po spojení rovnic 3.28 a 3.29 získáváme výsledný tvar

$$\vartheta_j = \vartheta_j - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)\boldsymbol{x}_j^{(i)}, \tag{3.30}$$

pokud bychom výslednou rovnici 3.30 rozepsali pro první dva parametry ϑ_0 a ϑ_1 dostali bychom

$$\vartheta_0 = \vartheta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)\boldsymbol{x}_0^{(i)}, \quad \boldsymbol{x}_0^{(i)} = 1 \tag{3.31}$$

$$\vartheta_1 = \vartheta_1 - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)\boldsymbol{x}_1^{(i)} \tag{3.32}$$

$$\vdots$$

3.7 Features scaling

V této podkapitole si řekneme něco o velikosti příznaků. První a nejdůležitější věcí je normování velikostí příznaků na stejnou velikost. Neboli, pokud použijeme opět náš příklad nákupu bytu a definujeme si velikost bytu v dm^2 (příznak x_1) a počet místností bytu (příznak x_2) a řekněme že:

$$\begin{aligned} x_1 &\in \langle 5000, 20000 \rangle, \\ x_2 &\in \langle 1, 5 \rangle, \end{aligned}$$

tak bude pro gradientní metodu velmi obtížné a více výpočetně náročné (bude nutno spočítat více iterací) nalézt minimum. Proto je důležité, aby všechny příznaky měli stejné škálování. V našem případě by to znamenalo

$$\begin{aligned} x_1 &= \frac{\text{velikost bytu}}{20000}, \\ x_2 &= \frac{\text{počet místností}}{5}. \end{aligned}$$

Více obecně lze říci, že se většinou volí normování na interval $\langle -1, 1 \rangle$.

Mean normalization

Velice častá normalizace střední hodnotou μ, konkrétně pro i-tý příznak

$$x_i \leftarrow \frac{x_i - \mu_i}{max - min} \tag{3.33}$$

nebo

$$x_i \leftarrow \frac{x_i - \mu_i}{std} \tag{3.34}$$

(std = Standard deviation).

3.8 Analytické řešení

V této sekci si ukážeme jak lze vypočítat minimum naší ztrátové funkce $J(\boldsymbol{\Theta})$ analyticky.

Příklad:

Předpokládejme 1D příklad ($\vartheta \in \mathcal{R}$), kdy máme ztrátovou funkci $J(\vartheta)$, která je funkcí jedné reálné proměnné

$$J(\vartheta) = a\vartheta^2 + b\vartheta + c \tag{3.35}$$

Pro minimalizaci naší kvadratické ztrátové funkce využijeme hledání extrému (položíme derivaci rovnou nule)

$$\frac{\mathrm{d}}{\mathrm{d}\vartheta} J(\vartheta) = \ldots \equiv 0 \tag{3.36}$$

a vyřešíme pro parametr ϑ.

Více obecněji pro $\boldsymbol{\Theta} \in \mathcal{R}^{n+1}$ bude naše ztrátová funkce nabývat tvaru

$$J(\vartheta_0,\ \vartheta_1,\ \ldots,\ \vartheta_m) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)} \right)^2 \tag{3.37}$$

dále položíme parciální derivace podle jednotlivých parametrů rovny nule (pro všechny j)

$$\frac{\partial}{\partial \vartheta_j} = \ldots \equiv 0 \tag{3.38}$$

a vyřešíme pro parametry $\vartheta_0,\ \vartheta_1,\ \ldots,\ \vartheta_m$.

Pro složení obyčejné rovnice budeme potřebovat matici $\boldsymbol{X} \in \mathcal{R}^{m\times(n+1)}$, která v řádcích obsahuje jednotlivé vektory $\left(\boldsymbol{x}^{(i)}\right)^\top$ a výstupní sloupcový vektor $\boldsymbol{y} \in \mathcal{R}^m$. Kde m je počet trénovacích vzorků (velikost trénovací množiny) a n je počet příznaků, kde $\boldsymbol{x}_0^{(i)} = 0$.

Nyní můžeme zapsat předpis pro výpočet parametrů $\boldsymbol{\Theta}$ analyticky

$$\boldsymbol{\Theta} = \left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top \boldsymbol{y}. \tag{3.39}$$

Pro analytické řešení není nutné příznaky normovat, jako v případě gradientního algoritmu, na stejné měřítko, jak je popsáno v sekci 3.7.

3.9 Gradientní algoritmus vs. analytické řešení

V této sekci budou zmíněny výhody a nevýhody gradientního algoritmu versus analytického řešení. Velikost trénovací množiny m a počet příznaků n.

Gradientní algoritmus	Analytické řešení
Nutno zvolit konstantu učení α.	Není nutné volit konstantu učení α.
Potřebuje mnoho iterací.	Nepotřebuje iterovat.
Pracuje dobře, když je n velké.	Pomalé pokud je n velké.
	Nutnost vypočítat inverzi $\left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1}$. (Složitost $\mathcal{O}\left(n^3\right)$).
Zvolit pokud $n > 10^6$.	Zvolit pokud $n < 10^4$.

Kapitola 4

Binární regrese - klasifikace

??????????????

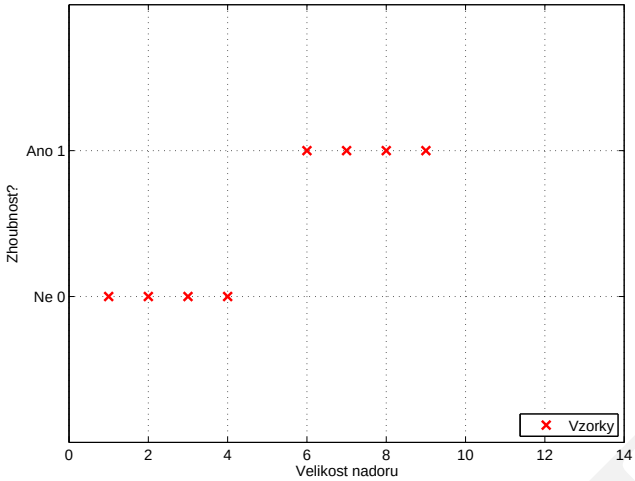
4.1 Klasifikace

Na úvod si uveďme několik klasifikačních příkladů:

Spam: je spam / není spam ?
Online transakce: podvodné ano / ne ?
Nádor: zhoubný / nezhoubný?

Z příkladů je zřejmé, že se jedná o rozhodnutí mezi dvěma možnostmi, tedy $y \in \{0, 1\}$, kde 0 je „negativní třída“ (nezhoubný nádor) a 1 je pozitivní třída (zhoubný nádor). Pokud máme $y \in \{0, 1\}$ jedná se o klasifikaci do dvou tříd, dále existuje klasifikace do více tříd, kde $y \in \{0, 1, 2, 3, \ldots\}$. V následujících kapitolách se budeme zabývat klasifikací do dvou tříd.

Nyní se dostáváme k problému jak vyvinout klasifikační algoritmus, což si můžeme ukázat na příkladu trénovací sady pro úlohu klasifikace nádorů. Nádor může být buď zhoubný nebo nezhoubný, tedy buďto 1 nebo 0. Dostupná data pro náš příklad jsou na obrázku 4.1.



Obrázek 4.1: Trénovací data.

Jedna z věcí, kterou bychom mohli udělat, je aplikace algoritmu lineární regrese, který známe z kapitoly 3. Výsledek aplikace hypotézy

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \boldsymbol{\Theta}^{\top}\boldsymbol{x} \tag{4.1}$$

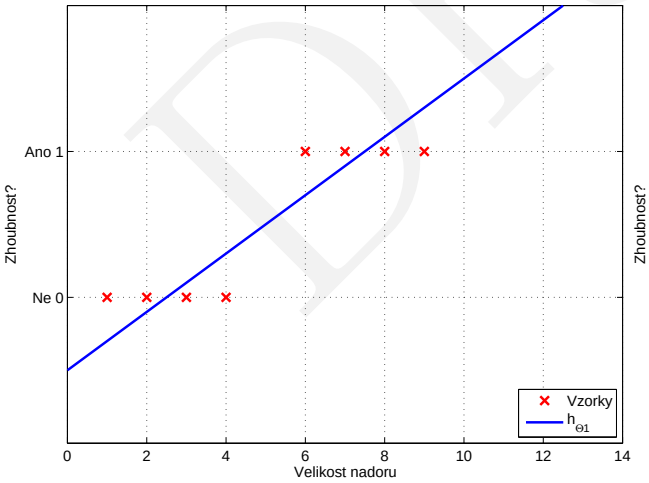
lze vidět na Obr. 4.2.

Pokud bychom chtěli udělat predikci ze získané hypotézy, tak můžeme využít klasifikátoru na základě prahování. Tedy hodnotu po jejíž překročení vzorek spadá již do druhé třídy, v případě binární regrese je to hodnota 0.5. Chceme tedy určit

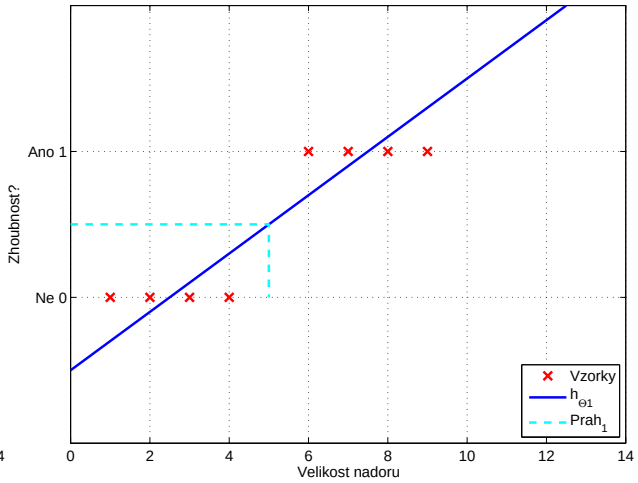
$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) \geq 0.5, \quad \text{predikuj } y = 1, \tag{4.2}$$

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) < 0.5, \quad \text{predikuj } y = 0. \tag{4.3}$$

Tento poznatek je reprezentován na Obr. 4.3.



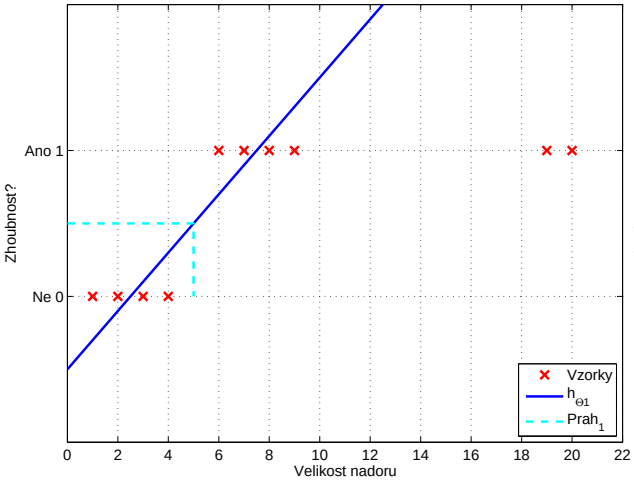
Obrázek 4.2: Lineární regrese.



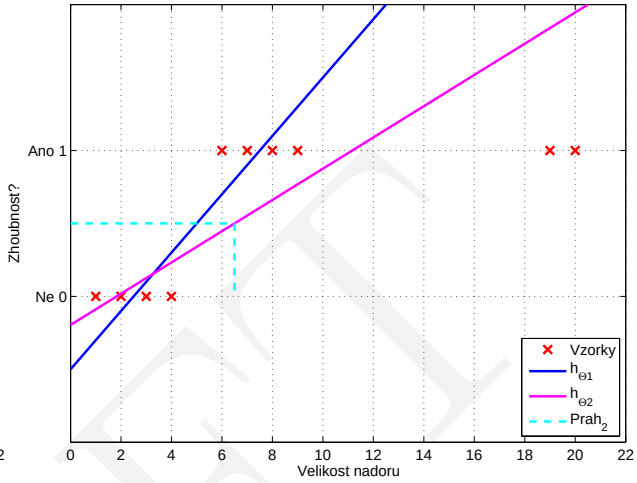
Obrázek 4.3: Lineární regrese - určení rozhodovacího prahu.

V tomto příkladě jsme si ukázali, že lineární regrese dělá něco co by se dalo považovat za řešení klasifikačního problému. Na následujícím rozšíření našeho příkladu bude ukázáno, že tomu tak není.

Rozšíříme náš příklad o další vzorky viz Obr. 4.4, kde nám přibyly dva vzorky více vpravo. Po aplikování lineární regrese, na rozšířená data, získáváme výsledek, který lze vidět na Obr. 4.5. Aktuální hypotéza $h_{\Theta 2}$ již nesplňuje podmínky definované vztahy 4.2 a 4.3, jelikož první vzorek, který má být vyhodnocen jako zhoubný by v tomto případě byl vyhodnocen jako nezhoubný, což může zásadně ovlivnit něčí život.



Obrázek 4.4: Rozšíření trénovací množiny dat o dva vzorky vpravo.



Obrázek 4.5: Lineární regrese - určení rozhodovacího prahu a zobrazení jeho selhání při predikci.

V tomto příkladě, lépe řečeno protipříkladě jsme si ukázali, proč je nevhodné využívat lineární regresi jako klasifikační algoritmus. Ve většině případů nefunguje správně. Pro klasifikaci na základě lineární regrese můžeme získat výsledek hypotézy $h_{\boldsymbol{\Theta}} > 1$ nebo $h_{\boldsymbol{\Theta}} < 0$, přičemž výsledek binární klasifikace je pouze $y = 0$ nebo $y = 0$. Proto bude nutné v dalších kapitolách přijít se sofistikovanějším postupem binární regrese (klasifikace), která splňuje podmínku $0 \leq h_{\boldsymbol{\Theta}} \leq 1$.

4.2 Reprezentace hypotézy

V této kapitole si ukážeme, jak správně reprezentovat hypotézu, tedy jakou funkci je vhodné použít v případě problému klasifikace.

V sekci 4.1 jsme zjistili, že bychom si pro binární regresní model přáli, aby platilo

$$0 \leq h_{\boldsymbol{\Theta}}(\boldsymbol{x}) \leq 1, \tag{4.4}$$

tedy aby výstupní hodnota byla mezi 0 a 1 (včetně). Pro reprezentaci hypotézy lineární regrese platí

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \boldsymbol{\Theta}^{\top}\boldsymbol{x}, \tag{4.5}$$

tuto hypotézu upravíme na tvar

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = g\left(\boldsymbol{\Theta}^{\top}\boldsymbol{x}\right), \tag{4.6}$$

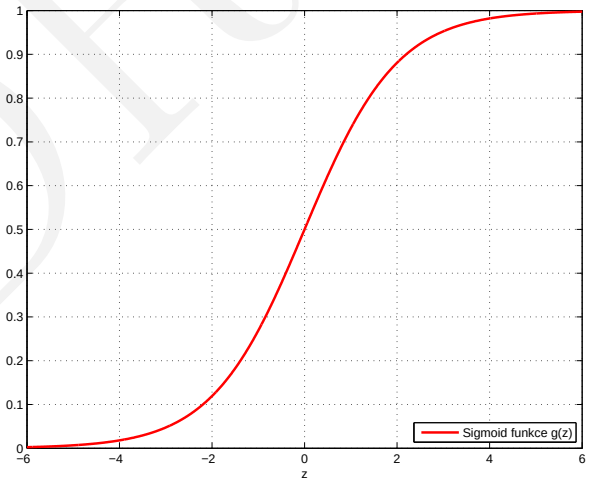
kde funkce g je definována následovně

$$g(\boldsymbol{z}) = \frac{1}{1+e^{-\boldsymbol{z}}}, \tag{4.7}$$

tato funkce se nazývá sigmoid funkce (Sigmoid function) nebo logistická/binární funkce (Logistic function). Jméno logistická funkce je důvod, proč vznik termín logistická/binární regrese. Pokud spojíme rovnice 4.6 a 4.7 získáme tvar

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \frac{1}{1+e^{-\boldsymbol{\Theta}^{\top}\boldsymbol{x}}} \tag{4.8}$$

a průběh grafu funkce $g(\boldsymbol{z})$ lze vidět na Obr. 4.6.



Obrázek 4.6: Sigmoidní funkce $g(z)$.

Poznamenejme, že

$$\begin{aligned} \lim_{z\to+\infty} \quad g(z) \quad &= 1, \\ \lim_{z\to-\infty} \quad g(z) \quad &= 0. \end{aligned}$$

Nyní je potřeba, stejně jako dříve, přizpůsobit parametry $\boldsymbol{\Theta}$ našim datům, tedy pomocí trénovací množiny zvolit hodnoty parametrů $\boldsymbol{\Theta}$ a využít tyto parametry pro výpočet predikce.

Dále si povíme něco o interpretaci výstupu hypotézy. Hypotéza $h_{\boldsymbol{\Theta}}(\boldsymbol{x})$ = odhadnutá pravděpodobnost, že výstup $y = 1$ na základě vstupních dat $\boldsymbol{x}$.

Příklad

Pokud $\boldsymbol{x} = [x_0,\ x_1]^\top = [1, \text{velikost nádoru}]^\top$ a $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = 0.7$, což říká pacientovi, že šance, že je nádor zhoubný je 70%. Nyní naše tvrzení zapíšeme formálněji

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = P(y = 1|\boldsymbol{x}; \boldsymbol{\Theta}), \tag{4.9}$$

neboli, pravděpodobnost, že $y = 1$, „na základě dat $\boldsymbol{x}$“, parametrizovaná parametry $\boldsymbol{\Theta}$. V případě klasifikace víme, že výstup y může nabývat pouze hodnot 0 nebo 1, proto platí

$$P(y = 0|\boldsymbol{x}; \boldsymbol{\Theta}) + P(y = 1|\boldsymbol{x}; \boldsymbol{\Theta}) = 1, \tag{4.10}$$

neboli

$$P(y = 0|\boldsymbol{x}; \boldsymbol{\Theta}) = 1 - P(y = 1|\boldsymbol{x}; \boldsymbol{\Theta}). \tag{4.11}$$

Poznámka

Pravděpodobnost může nabývat hodnot $0 - 1$, neboli $0 - 100\%$.

4.3 Rozhodovací hranice

V této kapitole si povíme něco o rozhodovací hranici, abychom získali větší ponětí o výpočtu hypotézy binární regrese (klasifikace).

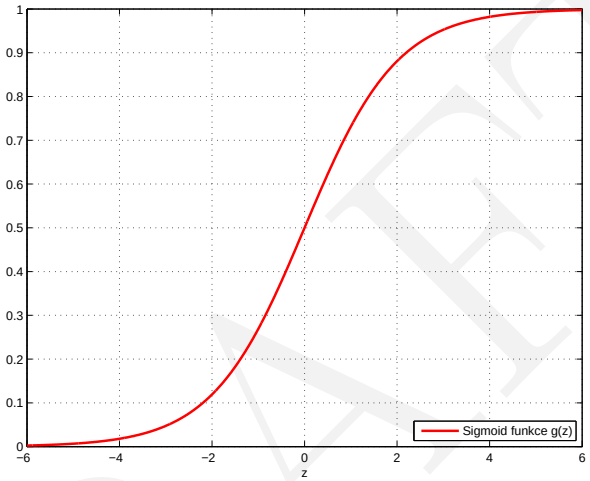
V kapitole 4.2 jsme si ukázali tvar hypotézy (sigmoid funkce)

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = g\left(\boldsymbol{\Theta}^{\top}\boldsymbol{x}\right), \tag{4.12}$$

kde

$$g(\boldsymbol{z}) = \frac{1}{1+e^{-\boldsymbol{z}}} \tag{4.13}$$

a graf lze vidět na Obr. 4.7.



Obrázek 4.7: Sigmoidní funkce $g(z)$.

Podrobněji vysvětlíme, kdy naše hypotéza bude predikovat na výstupu 0 nebo 1. Konkrétně naše hypotéza bude na výstupu predikovat $y = 1$ v případě

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = g\left(\boldsymbol{\Theta}^{\top}\boldsymbol{x}\right) = P(y = 1|\boldsymbol{x}; \boldsymbol{\Theta}). \tag{4.14}$$

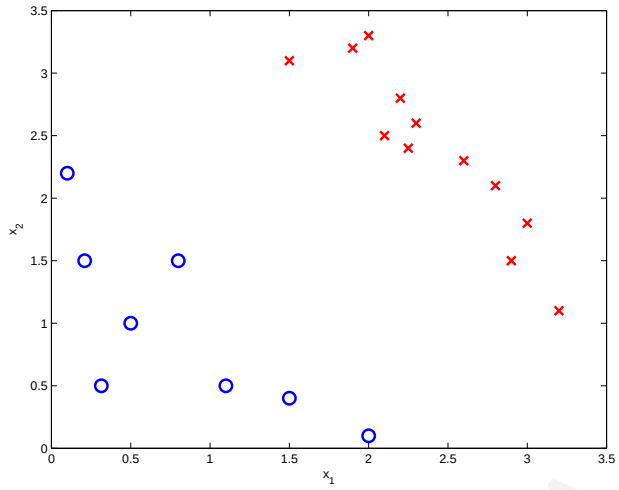
Předpokládejme, že klasifikátor predikuje výstup $y = 1$ pokud $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) \geq 0.5$, což platí pro $\boldsymbol{\Theta}^{\top}\boldsymbol{x} \geq 0$ a klasifikátor predikuje výstup $y = 0$ pokud $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) < 0.5$, což platí pro $\boldsymbol{\Theta}^{\top}\boldsymbol{x} < 0$.

Příklad - lineárně separabilní třídy

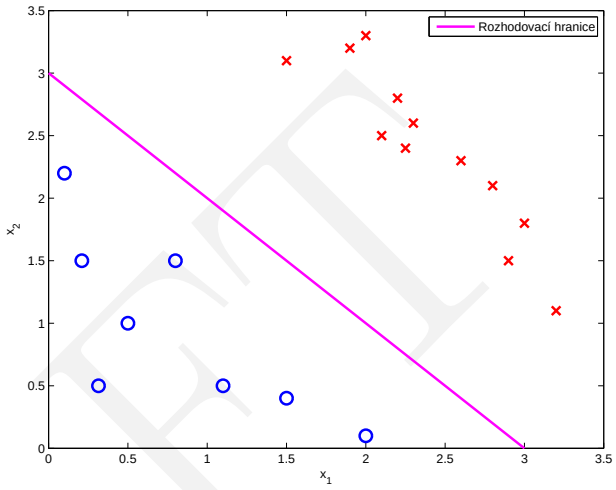
Předpokládejme, že máme trénovací sadu, která je znázorněna na Obr. 4.8 a naše hypotéza má tvar

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = g(\vartheta_0 + \vartheta_1 x_1 + \vartheta_2 x_2). \tag{4.15}$$

Nyní se nebudeme zabývat tím, jak přizpůsobit parametry tomuto modelu. Předpokládejme, že $\boldsymbol{\Theta} = [\vartheta_0,\ \vartheta_1,\ \vartheta 2]^\top = [-3,\ 1,\ 1]^\top$.



Obrázek 4.8: Vizualizace dvou tříd.



Obrázek 4.9: Vizualizace rozhodovací hranice.

Na příkladu si ukážeme, kdy hypotéza bude predikovat, jako výstup 0 a kdy 1. Hypotéza bude predikovat výstupní hodnotu $y = 1$ pokud

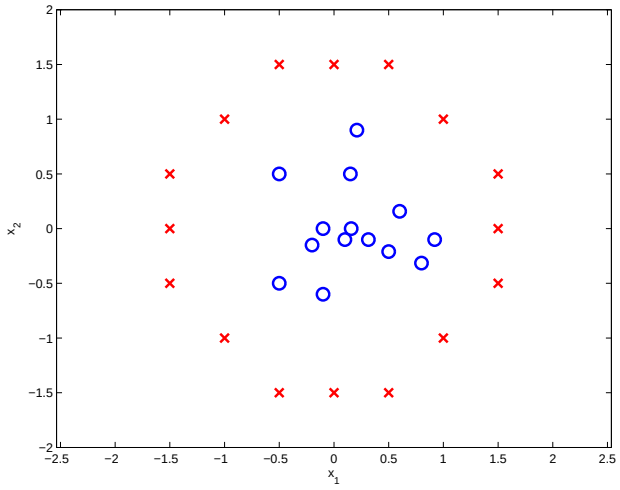
$$-3 + x_1 + x_2 \geq 0 \tag{4.16}$$

($\boldsymbol{\Theta}^\top \boldsymbol{x} \geq 0$), jinými slovy pro všechna x_1 a x_2, které splňují rovnici 4.16 bude naše hypotéza predikovat výstup $y = 1$. Řešení rovnice 4.16 je naznačeno na Obr. 4.9 a odpovídají mu červené křížky nad rozhodovací hranicí.

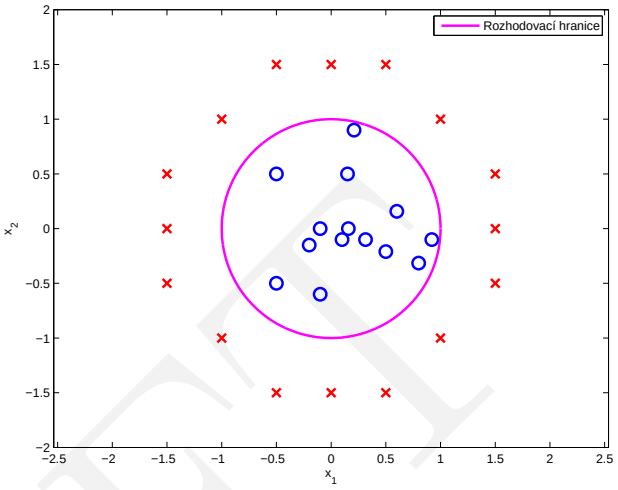
V případě, kdy $x_1 + x_2 = 3$, tak to odpovídá přesně $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = 0.5$. Jinak řečeno řešení odpovídá přesně rozhodovací hranici mezi třídami.

Příklad - lineárně neseparabilní třídy

Předpokládejme složitější příklad, kde nelze jednoduše použít binární regresi (lineární rozhodovací hranici) a musíme využít polynomiální regrese, tedy musíme využít vyšší stupně polynomů jako příznaky viz Obr. 4.10.



Obrázek 4.10: Vizualizace lineárně neseparabilních tříd.



Obrázek 4.11: Vizualizace lineárně neseparabilních tříd společně s rozhodovací hranicí.

Naše hypotéza má tvar $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = g(\vartheta_0 + \vartheta_1 x_1 + \vartheta_2 x_2 + \vartheta_3 x_1^2 + \vartheta_4 x_2^2)$, kde jsme oproti předcházejícímu příkladu (rovnice 4.15) přidali dva příznaky x_1^2 a x_2^2 a máme 5 parametrů $\vartheta_{0,\ldots,4}$. Nebudeme zabývat tím, jak vypočítat parametry $\boldsymbol{\Theta}$ (bude popsáno v následujících sekcích), řekněme, že $\boldsymbol{\Theta} = [\vartheta_0,\ \vartheta_1,\ \vartheta_2,\ \vartheta_3,\ \vartheta_4]^\top = [-1,\ 0,\ 0,\ 1,\ 1]^\top$. Což znamená, že naše hypotéza bude predikovat výstup $y = 1$ pokud $-1 + x_1^2 + x_2^2 \geq 0$ (pro směr ven od rozhodovací hranice), tato rovnice odpovídá rovnici kružnice $x_1^2 + x_2^2 = 1$ o poloměru $r = 1$. Rozhodovací hranice je vidět na Obr. 4.11.

Poznámka

Rozhodovací hranice je vlastnost hypotézy, konkrétně parametrů $\boldsymbol{\Theta}$, není to vlastnost trénovací sady. Trénovací sada je pouze využita k nalezení parametrů hypotézy $\boldsymbol{\Theta}$.

4.4 Ztrátová funkce

V této sekci si povíme jak vypočítat parametry $\boldsymbol{\Theta}$ pro binární regresi / klasifikaci.

Definujme problém učení s učitelem pro binární regresní model. Mějme trénovací množinu

$$\left\{\left(\boldsymbol{x}^{(1)}, y^{(1)}\right), \left(\boldsymbol{x}^{(2)}, y^{(2)}\right), \ldots, \left(\boldsymbol{x}^{(m)}, y^{(m)}\right)\right\} \tag{4.17}$$

o velikosti m vzorků, kde

$$\boldsymbol{x}^{(i)} = \left[x_0,\ x_1,\ \ldots,\ x_n\right]^\top, \tag{4.18}$$

$\boldsymbol{x}^{(i)} \in \mathrm{R}^{n+1}$ a $x_0 = 1$, $y \in \{0, 1\}$. Naše hypotéza má tvar

$$h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}\right) = \frac{1}{1 + e^{-\boldsymbol{\Theta}^\top \boldsymbol{x}}}. \tag{4.19}$$

Nyní si položme zásadní otázku, jak zvolit parametry $\boldsymbol{\Theta}$ v závislosti na naší trénovací množině?

Připomeňme si ztrátovou funkci $J\left(\boldsymbol{\Theta}\right)$ pro lineární regresi (viz sekce 3.3), kterou zapíšeme v následujícím tvaru

$$J\left(\boldsymbol{\Theta}\right) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}\left(h_\Theta\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2, \tag{4.20}$$

dále můžeme ztrátovou funkci přepsat do tvaru

$$J\left(\boldsymbol{\Theta}\right) = \frac{1}{m}\sum_{i=1}^{m} Cena\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right), y^{(i)}\right), \tag{4.21}$$

kde

$$Cena\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right), y^{(i)}\right) = \frac{1}{2}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2. \tag{4.22}$$

Nyní lze z rovnice 4.21 lépe vidět, že ztrátová funkce je $\frac{1}{m}$ krát součet Cen přes trénovací sadu. Rovnici 4.22 lze dále zjednodušit vynecháním horních indexů

$$Cena\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}\right), y\right) = \frac{1}{2}\left(h_\Theta\left(\boldsymbol{x}\right) - y\right)^2 \tag{4.23}$$

a říká nám jakou $Cenu$ zaplatíme pokud výstupní hodnota hypotézy bude $h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}\right)$ a výstupní informace od učitele bude hodnota y.

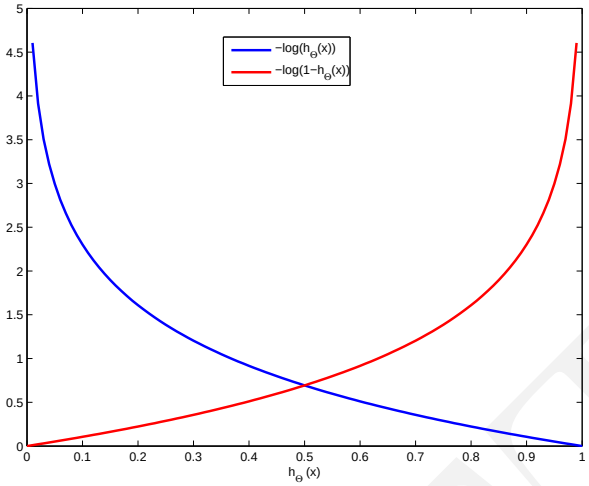
Tento předpis 4.22 pro výpočet $Ceny$ platí pro lineární regresi. Avšak v této sekci se zajímáme o binární regresi. Pokud bychom chtěli minimalizovat tuto ztrátovou funkci v rovnici 4.21, tak tato funkce může být nekonvexní vzhledem k parametrům $\boldsymbol{\Theta}$ (není zaručeno, že dosáhneme globálního minima, jako v případě konvexní ztrátové funkce).

Uveďme si ztrátovou funkci, která se využívá při binární regresi

$$Cena\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}\right), y\right) = \begin{cases} -\log\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}\right)\right), & \text{pro } y = 1 \\ -\log\left(1 - h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}\right)\right), & \text{pro } y = 0 \end{cases} \tag{4.24}$$

jinými slovy lze říci, že se jedná jakou ztrátu (kolik nás stojí) utrpíme pokud $y = 0$ nebo $y = 1$.

Nyní si ztrátovou funkci pro binární regresi, z rovnice 4.24, rozebereme podrobněji. Na Obr. 4.12 lze vidět průběh ztrátové funkce.



Obrázek 4.12: Průběh ztrátové funkce.

Pokud bude $y = 1$ a naše hypotéza $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = 1$, tak utrpíme ztrátu rovnu $Cena = 0$. Ale pokud se bude hodnota hypotézy zmenšovat $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) \to 0$, tak se utrpěná ztráta bude zvětšovat $Cena \to \infty$. V druhém případě pokud $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = 0$ (predikce $P(y = 1|\boldsymbol{x}; \boldsymbol{\Theta}) = 0$), ale $y = 1$, tak utrpěná ztráta bude velmi velká.

Pro $y = 0$ platí obdobně druhá část rovnice 4.24.

4.5 Zjednodušená ztrátová funkce a gradientní metoda

Naše ztrátová funkce $J(\boldsymbol{\Theta})$ ve tvaru

$$J(\boldsymbol{\Theta}) = \frac{1}{m}\sum_{i=1}^{m} Cena\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right), y^{(i)}\right), \tag{4.25}$$

kde

$$Cena\left(h_{\boldsymbol{\Theta}}(\boldsymbol{x}), y\right) = \begin{cases} -\log\left(h_{\boldsymbol{\Theta}}(\boldsymbol{x})\right), & \text{pro } y = 1 \\ -\log\left(1 - h_{\boldsymbol{\Theta}}(\boldsymbol{x})\right), & \text{pro } y = 0 \end{cases} \tag{4.26}$$

poznamenejme že $y = 0$ nebo $y = 1$ (pokaždé, žádná jiná možnost není).

Naší *Cenu* z rovnice 4.26 můžeme zapsat v jedné rovnici, která má tvar

$$Cena\left(h_{\boldsymbol{\Theta}}(\boldsymbol{x}), y\right) = -y\log\left(h_{\boldsymbol{\Theta}}(\boldsymbol{x})\right) - (1-y)\log\left(1 - h_{\boldsymbol{\Theta}}(\boldsymbol{x})\right), \tag{4.27}$$

pokud $y = 1$, tak rovnice 4.27 přechází na první část rovnice 4.26 a pokud $y = 0$, tak rovnice 4.27 přechází na druhou část rovnice 4.26.

Po spojení a úpravě rovnic 4.27 a 4.25 získáme výsledný tvar

$$J(\boldsymbol{\Theta}) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)}\log\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right)\right) + \left(1 - y^{(i)}\right)\log\left(1 - h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right)\right)\right]. \tag{4.28}$$

Cílem je nalézt takové parametry $\boldsymbol{\Theta}$, které minimalizují ztrátovou funkci J

$$\min_{\boldsymbol{\Theta}} J(\boldsymbol{\Theta}), \tag{4.29}$$

a následně pro vypočítáme predikci naší hypotézy (na základě nového vstupu x)

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{\Theta}^{\top}\boldsymbol{x}}}. \tag{4.30}$$

Nyní můžeme konečně zapsat celý gradientní algoritmus, opakuj

$$\vartheta_j = \vartheta_j - \alpha\frac{\partial}{\partial\vartheta_j}J(\boldsymbol{\Theta}), \tag{4.31}$$

po každé iteraci aktualizuj všechny parametry ϑ_j najednou dokud algoritmus nedokonverguje.

Poznámka

V rovnici 4.31 bude využit tvar

$$\frac{\partial}{\partial\vartheta_j}J(\boldsymbol{\Theta}) = \frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)\boldsymbol{x}_j^{(i)}. \tag{4.32}$$

a proto je výhodnější psát rovnici 4.31 ve tvaru

$$\vartheta_j = \vartheta_j - \alpha\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)\boldsymbol{x}_j^{(i)}. \tag{4.33}$$

Algoritmus vypadá totožně jako v případě lineární regrese (viz 3.30) s rozdílnou hypotézou $h_{\boldsymbol{\Theta}}(\boldsymbol{x})$, v případě lineární regrese

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \boldsymbol{\Theta}^{\top}\boldsymbol{x} \tag{4.34}$$

a v případě binární regrese se jedná o

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{\Theta}^{\top}\boldsymbol{x}}}. \tag{4.35}$$

Poznámka

Při výpočtu parametrů $\boldsymbol{\Theta}$ pomocí gradientního algoritmu v případě binární regrese je také vhodné využít Features scaling, jako v případě lineární regrese (viz sekce 3.7).

4.6 Pokročilé algoritmy optimalizace

V této sekci si uvedeme příklady několika pokročilejších algoritmů optimalizace, které jsou lépe využitelné pro velké problémy strojového učení, kde je například velký počet příznaků.

Připomeňme náš optimalizační problém, máme ztrátovou funkci $J(\boldsymbol{\Theta})$ a chceme jí minimalizovat

$$\min_{\boldsymbol{\Theta}} J(\boldsymbol{\Theta}). \tag{4.36}$$

Musíme tedy znát rovnice pro výpočet $J(\boldsymbol{\Theta})$ a $\frac{\partial}{\partial \vartheta_j} J(\boldsymbol{\Theta})$ následně můžeme využít zmíněný gradientní algoritmus, nebo nějakou z pokročilejších optimalizačních metod jako jsou například

Název algoritmu	Výhody	Nevýhody
Conjugate gradient	Nemusí se volit	Více složité
BFGS	konstanta učení α.	
L-BFGS	Rychlejší než gradientní metoda.	

Tabulka 4.1: Příklady pokročilejších optimalizačních algoritmů.

Příklad

Nyní si uvedeme příklad společně s jeho implementací v MATLABu. Mějme vektor parametrů $\boldsymbol{\Theta} = [\vartheta_1,\ \vartheta_2]^\top$ a ztrátovou funkci $J(\boldsymbol{\Theta}) = (\vartheta_1 - 5)^2 + (\vartheta_2 - 5)^2$, dále vypočteme parciální derivace ztrátové funkce

$$\frac{\partial}{\partial \vartheta_1} J(\boldsymbol{\Theta}) = 2 \cdot (\vartheta_1 - 5), \tag{4.37}$$

$$\frac{\partial}{\partial \vartheta_2} J(\boldsymbol{\Theta}) = 2 \cdot (\vartheta_2 - 5). \tag{4.38}$$

Program v prostředí MATLAB by mohl vypadat následovně, nejprve je nutné si vytvořit funkci pro výpočet ztrátové funkce, která bude vracet hodnotu ztrátové funkce a jednotlivé gradienty.

```
1 function [jVal, grad] = costFunction(theta)
2
3 jVal = (theta(1, 1) - 5)^2 + (theta(2, 1) - 5)^2;
4
5 grad = zeros(size(theta, 1), 1);
6 grad(1, 1) = 2 * (theta(1, 1) - 5);
7 grad(2, 1) = 2 * (theta(2, 1) - 5);
8
9 end
```

Optimalizaci ztrátové funkce (hledání minima) můžeme spustit následovně.

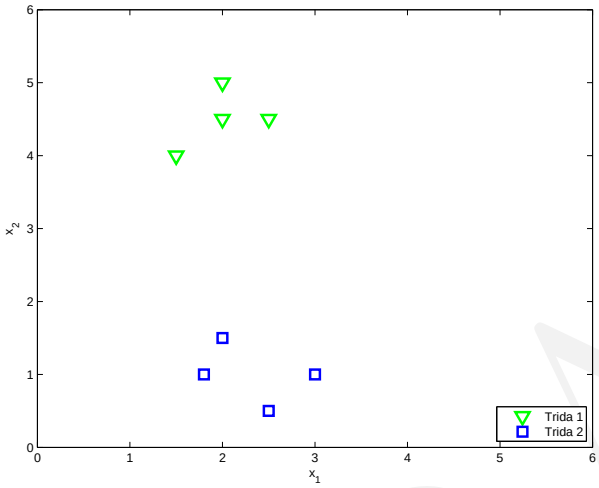
```
initialTheta = [1; 2];
options = optimset('GradObj', 'on', 'MaxIter', 100);
[optTheta, functionVal, exitFlag] = ...
    fminunc(@costFunction, initialTheta, options);
```

Výsledné hodnoty pro $\vartheta_{1,2} = 5$, což lze pro tento jednoduchý případ vidět z rovnic 4.37 a 4.38. Jak lze vidět z příkladu, nebylo nutné nastavovat parametr α, funkce *fminunc* v MATLABu si jej nastavila sama.

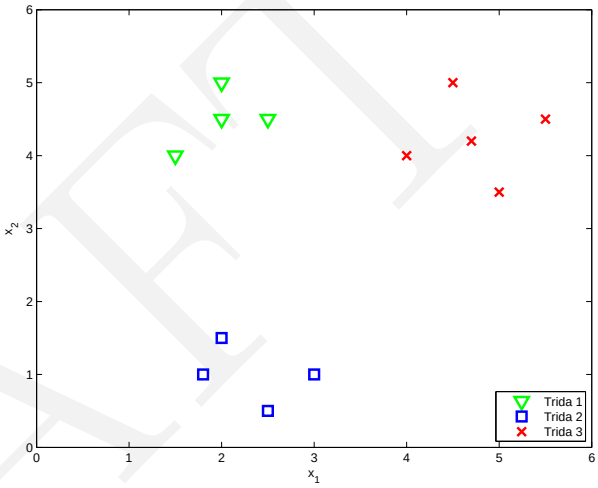
4.7 Klasifikace do více tříd

V této sekci si povíme o klasifikaci do více tříd. Tento problém lze uchopit tak, že budeme klasifikovat jednu třídu proti všem ostatním, náš klasifikační algoritmus by se dal nazvat jeden proti všem.

Nejdříve si popišme náš klasifikační problém, lze ho demonstrovat na příkladu klasifikace emailů. Představme si, že bychom rádi příchozí emaily rozdělovali do složek, nebo označovali podle skupiny lidí, od kterých nám email přišel, například pracovní emaily, emaily od kamarádů, emaily od rodiny a emaily od známých, s kterými sdílíme koníčky. Neboli chceme klasifikovat do tříd $y = \{1,\ 2,\ 3,\ 4\}$. Na Obr. 4.13 a 4.14 je vidět rozdíl v klasifikaci do dvou a více tříd.



Obrázek 4.13: Klasifikace do dvou tříd.



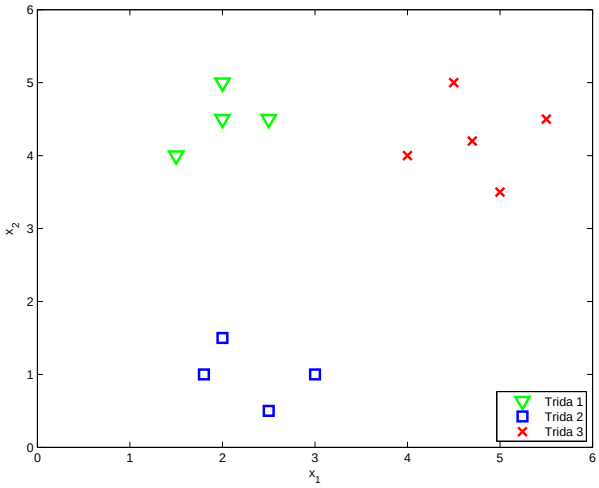
Obrázek 4.14: Klasifikace do více tříd.

Z předešlých sekcí víme jak klasifikovat do dvou tříd. Nyní tento přístup použijeme při řešení problému klasifikace do více tříd. Na Obr. 4.15 je znázorněno rozložení tříd a následně na Obr. 4.16 je ukázána klasifikace třídy 1 proti třídě, která je zkonstruována spojením všech ostatních tříd (v našem případě tříd 2 a 3). Na dalších Obr. 4.17 a 4.18 jsou naznačeny zbylé kombinace.Pro jednotlivé klasifikace je nutné vytvořit hypotézu

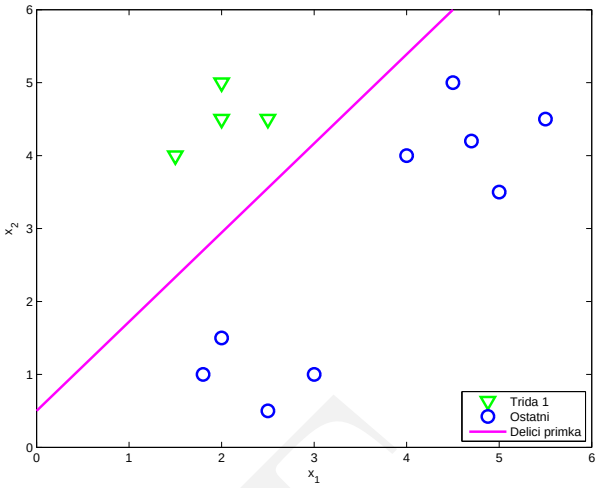
$$h_{\boldsymbol{\Theta}}^{(i)}(\boldsymbol{x}), \quad i \in \{1,\ 2,\ 3\}, \tag{4.39}$$

která odpovídá pravděpodobnostem

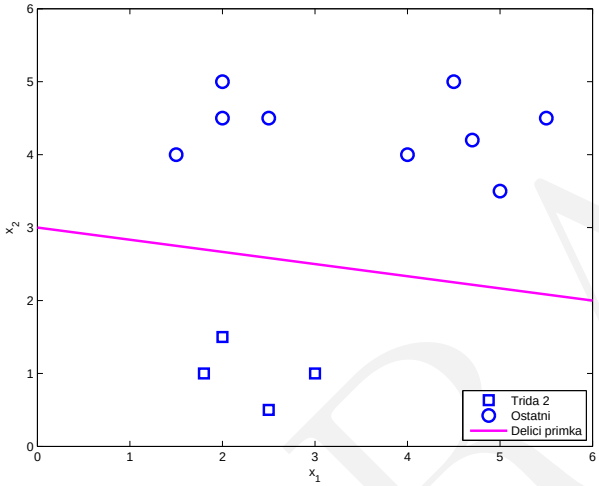
$$h_{\boldsymbol{\Theta}}^{(i)}(\boldsymbol{x}) = P(y = i|\boldsymbol{x}; \boldsymbol{\Theta}). \tag{4.40}$$



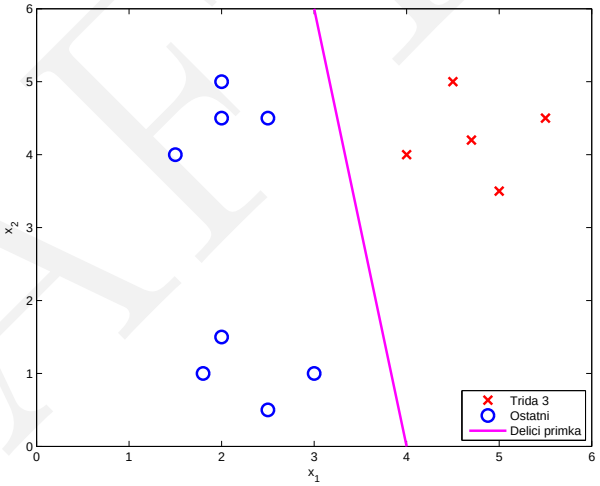
Obrázek 4.15: Rozložení tříd $y = \{1,\ 2,\ 3\}$.



Obrázek 4.16: Klasifikace třídy 1.



Obrázek 4.17: Klasifikace třídy 2.



Obrázek 4.18: Klasifikace třídy 3.

Shrnutí

Nejdříve je nutné natrénovat binární klasifikátory $h_{\boldsymbol{\Theta}}^{(i)}(\boldsymbol{x})$ pro všechny třídy i pro predikci pravděpodobnosti, že vzorek bude patřit do třídy $y = i$. Následně pro nový vstup x klasifikovat podle všech hypotéz a vybrat takovou třídu i, která maximalizuje pravděpodobnost hypotézy

$$\max_i h_{\boldsymbol{\Theta}}^{(i)}(\boldsymbol{x})\,. \tag{4.41}$$

Kapitola 5

Regularizace

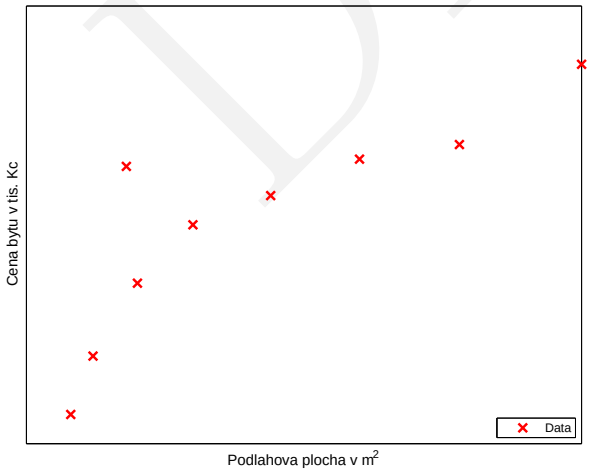
5.1 Problém přetrénování

Nyní když známe algoritmy lineární a binární regrese, které se hodí pro řešení mnoha úloh strojového učení, musíme si říci něco o problému zvaném přetrénování. Tento problém způsobuje, že dané algoritmy ve výsledku fungují velmi špatně.

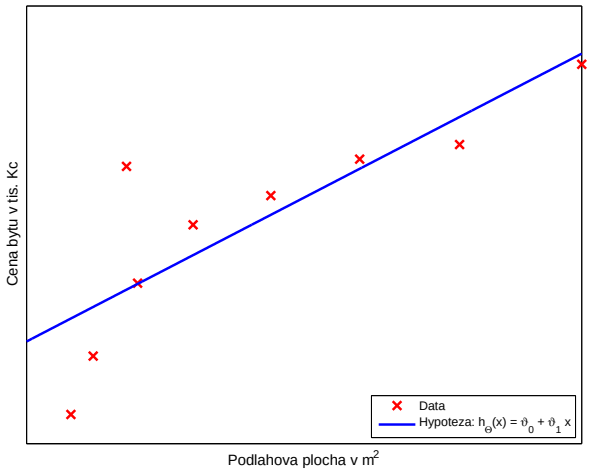
V této sekci si vysvětlíme v čem problém přetrénování spočívá a v následujících sekcích bude popsán postup jak se s daným problémem vypořádat.

Vraťme se k našemu problému predikce ceny velikosti bytu, ve kterém jsme využili algoritmu lineární regrese.

Na Obr. 5.1 jsou znázorněná trénovací data. Pokud využijeme lineární regresi společně s hypotézou $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \vartheta_0 + \vartheta_1 x$, tak získáme predikci ve tvaru, který lze vidět na Obr. 5.2. V tomto případě naše zvolená hypotéza $h_{\boldsymbol{\Theta}}(\boldsymbol{x})$ nerespektuje dostatečně tvar našich trénovacích dat. Přesnější anglický termín pro takto nastalou situaci zní *underfit* nebo *high bias*, což se dá vysvětlit jako problém, kdy se snažíme použít regresi na trénovací data s málo komplexní hypotézou. Tedy hypotéza nemůže dostatečně přesně vystihovat tvar dat (nezobecňuje).



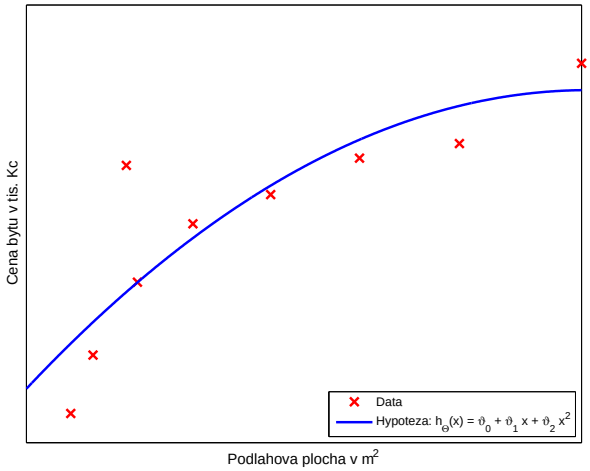
Obrázek 5.1: Trénovací data.



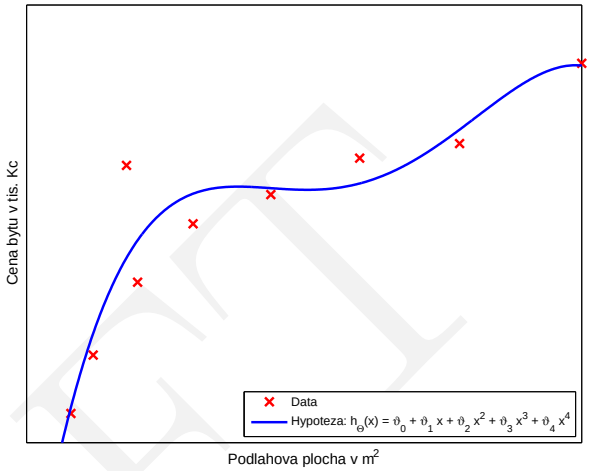
Obrázek 5.2: Lineární regrese (*underfit*, *high bias*).

Na Obr. 5.3 lze vidět regresi s hypotézou ve tvaru $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \vartheta_0 + \vartheta_1 x + \vartheta_2 x^2$, která velmi dobře zobecňuje trénovací data a proto by byla v tomto případě nejlepší volbou.

Dále na Obr. 5.4 lze vidět regresi s hypotézou $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \vartheta_0 + \vartheta_1 x + \vartheta_2 x^2 + \vartheta_2 x^3 + \vartheta_2 x^4$, jejíž chování je velmi „divoké“ a nezobecňuje trénovací data. V tomto případě se snaží hypotéza co nejvíce respektovat trénovací data za cenu ztráty obecnosti. Tento problém se nazývá přetrénování, anglicky *overfitting* nebo *high variance*.



Obrázek 5.3: Regrese pomocí hypotézy ve tvaru polynomu druhého stupně (*just right*).



Obrázek 5.4: Regrese pomocí hypotézy ve tvaru polynomu čtvrtého stupně (*overfitting*, *high variance*).

Přetrénování (*overfitting*):

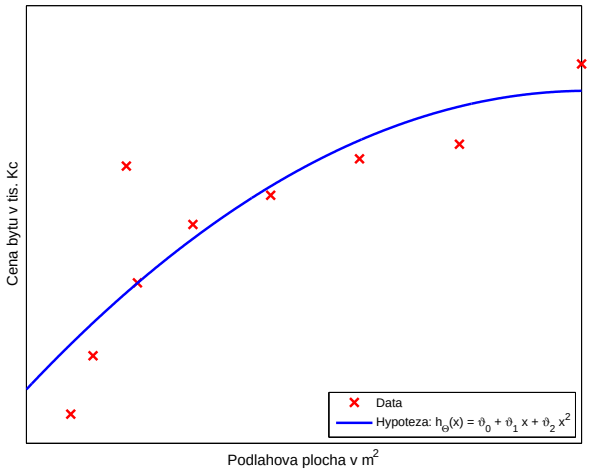
V případě, že máme příliš mnoho příznaků, hypotéza může velmi dobře respektovat trénovací data $\left(J(\boldsymbol{\Theta}) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2 \approx 0\right)$, ale nebude schopna správně klasifikovat nově příchozí vzorky. Nebude schopna generalizovat, tedy predikce na nově příchozích vzorcích selže.

Možnosti odstranění přetrénování:

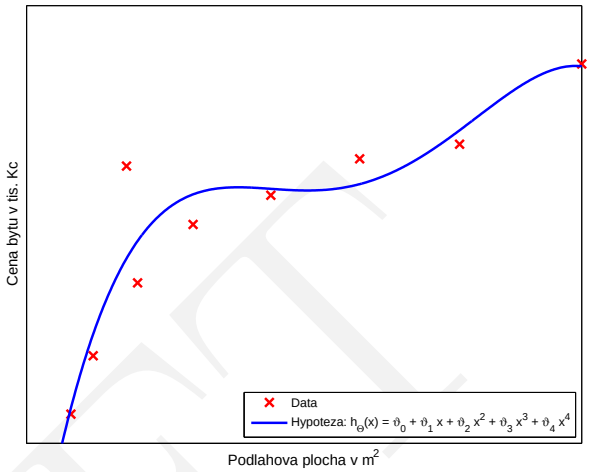
1. Redukovat počet příznaků.
   - Manuálně zvolit příznaky, které budou zachovány.
   - Automatická redukce příznaků například PCA.
2. Regularizace.
   - Zachovat všechny příznaky, ale redukovat velikost / váhu parametrů ϑ_j.
   - Funguje správně pokud máme velké množství příznaků, kde každý malou měrou přispívá k predikci y.

5.2 Ztrátová funkce

V sekci 5.1 jsme si ukázali jak lze pomocí kvadratické hypotézy $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \vartheta_0 + \vartheta_1 x + \vartheta_2 x^2$ dobře zobecnit naše trénovací data. Dále již víme, že hypotéza ve tvaru $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \vartheta_0 + \vartheta_1 x + \vartheta_2 x^2 + \vartheta_2 x^3 + \vartheta_2 x^4$ nezobecňuje naší trénovací množinu dat dobře.



Obrázek 5.5: Regrese pomocí hypotézy ve tvaru polynomu druhého stupně (*just right*).



Obrázek 5.6: Regrese pomocí hypotézy ve tvaru polynomu čtvrtého stupně (*overfitting*, *high variance*).

Předpokládejme, že budeme penalizovat parametry ϑ_3 a ϑ_4 (jejich hodnoty budou malé). V rovnici 5.1 je ukázána naše známá optimalizace

$$\min_{\boldsymbol{\Theta}} \frac{1}{2m} \sum_{i=1}^{m} \left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)} \right)^2, \tag{5.1}$$

kterou můžeme upravit na tvar

$$\min_{\boldsymbol{\Theta}} \frac{1}{2m} \sum_{i=1}^{m} \left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)} \right)^2 + 1000\vartheta_3^2 + 1000\vartheta_4^2, \tag{5.2}$$

kde hodnoty 1000 jsou pouze ilustrativní a reprezentují velká čísla (velké váhy). Tedy pokud bychom chtěli minimalizovat rovnici 5.2, tak to lze za předpokladu, že ϑ_3 a ϑ_4 budou malá čísla. V případě, že $\vartheta_3 \approx 0$ a $\vartheta_4 \approx 0$, tak získáme opět náš minimalizační problém jako v rovnici 5.1. Ještě větší dopad to má na tvar hypotézy $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \vartheta_0 + \vartheta_1 x + \vartheta_2 x^2 + \vartheta_2 x^3 + \vartheta_2 x^4$, která dříve vedla k přetrénování. Ta v případě, že $\vartheta_3 \approx 0$ a $\vartheta_4 \approx 0$ přejde na tvar $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \vartheta_0 + \vartheta_1 x + \vartheta_2 x^2$. Tímto se zbavíme problému přetrénování a získáme tvar hypotézy, který dobře zobecňuje trénovací data.

Regularizace

Čím jsou menší hodnoty parametrů $\vartheta_0, \vartheta_1, \ldots, \vartheta_n$, tím „jednodušší“ tvar hypotézy získáme a tím méně je tvar hypotézy náchylný k přetrénování. Zapsáno rovnicí

$$J(\boldsymbol{\Theta}) = \frac{1}{2m} \left(\sum_{i=1}^{m} \left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)} \right)^2 + \lambda \sum_{j=1}^{n} \vartheta_j^2 \right), \tag{5.3}$$

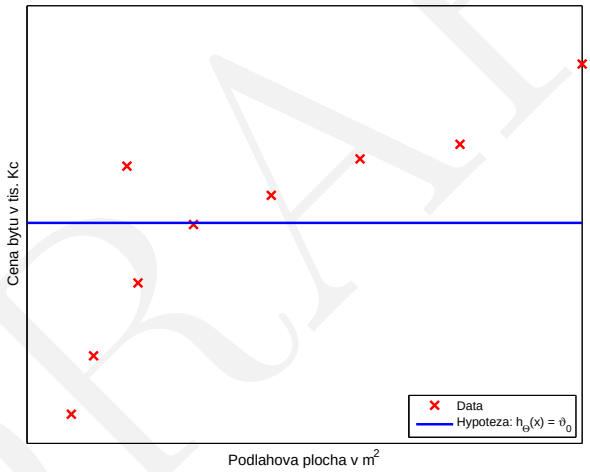
kde je nejdříve podotknout, že nová suma s indexem j má jinou mez n než první suma. To je způsobeno tím, že se regularizace vztahuje pouze na parametry $\vartheta_1, \vartheta_2, \ldots, \vartheta_m$, tedy parametr ϑ_0 není regularizován, protože je vždy násoben jedničkou, nepřijde do kontaktu s příznaky vzorků. Dále parametr λ je takzvaný regularizační parametr, který kontroluje vyvážení mezi tím, jak hodně bude regularizace ovlivňovat původní ztrátovou funkci $J(\boldsymbol{\Theta})$. Regularizace je dána vztahem

$$\lambda \sum_{j=1}^{n} \vartheta_j^2. \tag{5.4}$$

Našim cílem je jako dříve minimalizovat ztrátovou funkci

$$\min_{\boldsymbol{\Theta}} J(\boldsymbol{\Theta}) \tag{5.5}$$

Co se stane, pokud bude regularizační parametr λ příliš velký? Řekněme, že $\lambda = 10^{10}$. Poté hypotéza ve tvaru $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \vartheta_0 + \vartheta_1 x + \vartheta_2 x^2 + \vartheta_2 x^3 + \vartheta_2 x^4$ přejde na tvar $h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \vartheta_0$ a výsledný průběh lze vidět na Obr. 5.7



Obrázek 5.7: Použití regularizace k získání *underfit* hypotézy.

Z uvedeného vyplývá, že je nutné volit regularizační parametr λ velmi opatrně. S velkým λ dosáhneme *underfit* problému a s malým λ neodstraníme *overfit* problém.

5.3 Regularizace lineární regrese

Pro lineární regresi máme odvozen gradientní algoritmus a analytické řešení. V této sekci oba algoritmy zobecníme s využitím regularizace.

Regularizace gradientního algoritmu lineární regrese

Máme naší optimalizační úlohu lineární regrese společně s regularizací

$$J(\boldsymbol{\Theta}) = \frac{1}{2m}\left(\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{n}\vartheta_j^2\right), \tag{5.6}$$

snažíme se minimalizovat ztrátovou funkci

$$\min_{\boldsymbol{\Theta}} J(\boldsymbol{\Theta}) \tag{5.7}$$

Připomeňme si dříve představený gradientní algoritmus lineární regrese. Opakuj

$$\vartheta_j = \vartheta_j - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)x_j^{(i)}, \quad j = 0, 1, \ldots, n. \tag{5.8}$$

Tento algoritmus je možno rozdělit do dvou kroků. Rozdělit výpočet pro ϑ_0 a $\vartheta_{1,\ldots,n}$. Opakuj

$$\vartheta_0 = \vartheta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)x_0^{(i)}, \tag{5.9}$$

$$\vartheta_j = \vartheta_j - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)x_j^{(i)}, \quad j = 1, 2, \ldots, n. \tag{5.10}$$

Doposud na algoritmu nebylo nic změněno, pouze byl rozepsán do dvou kroků. Protože víme, že na parametr ϑ_0 se regularizace nevztahuje, proto můžeme algoritmus pozměnit a přidat regularizační člen. Následně bude mít gradientní algoritmus tvar. Opakuj

$$\vartheta_0 = \vartheta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)x_0^{(i)}, \tag{5.11}$$

$$\vartheta_j = \vartheta_j - \alpha\left(\frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)x_j^{(i)} + \frac{\lambda}{m}\vartheta_j\right), \quad j = 1, 2, \ldots, n, \tag{5.12}$$

kde platí

$$\frac{\partial}{\partial\vartheta_0} J(\boldsymbol{\Theta}) = \left(\frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)x_0^{(i)}\right), \tag{5.13}$$

$$\frac{\partial}{\partial\vartheta_j} J(\boldsymbol{\Theta}) = \left(\frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)x_j^{(i)} + \frac{\lambda}{m}\vartheta_j\right), \quad j = 1, 2, \ldots, n. \tag{5.14}$$

Rovnice 5.12 lze přepsat do tvaru

$$\vartheta_j = \vartheta_j\left(1 - \alpha\frac{\lambda}{m}\right) - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)x_j^{(i)}, \tag{5.15}$$

kde platí

$$1 - \alpha\frac{\lambda}{m} < 1. \tag{5.16}$$

Regularizace analytického řešení lineární regrese

Připomeňme si

$$\boldsymbol{X} = \left[\left(\boldsymbol{x}^{(1)}\right)^\top, \ldots, \left(\boldsymbol{x}^{(m)}\right)^\top\right]^\top, \quad \boldsymbol{X} \in \mathcal{R}^{m\times(n+1)} \tag{5.17}$$

a

$$\boldsymbol{y} = \left[y^{(1)}, \ldots, y^{(m)}\right]^\top, \quad \boldsymbol{y} \in \mathcal{R}^{m}. \tag{5.18}$$

Analytické řešení pro výpočet $\boldsymbol{\Theta}$ má tvar

$$\boldsymbol{\Theta} = \left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top y. \tag{5.19}$$

Cílem je minimalizovat ztrátovou funkci

$$\min_{\boldsymbol{\Theta}} J\left(\boldsymbol{\Theta}\right). \tag{5.20}$$

Pokud využijeme regularizaci, tak naše analytické řešení přejde na tvar

$$\boldsymbol{\Theta} = \left(\boldsymbol{X}^\top \boldsymbol{X} + \lambda \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}\right)^{-1} \boldsymbol{X}^\top y \tag{5.21}$$

kde nová matice má velikost $\mathcal{R}^{(n+1)\times(n+1)}$

Rozšíření

Předpokládejme, že $m \leq n$, kde m je počet vzorků a n je počet příznaků a

$$\boldsymbol{\Theta} = \left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top y, \tag{5.22}$$

pak je matice $\boldsymbol{X}^\top \boldsymbol{X}$ neinvertovatelná / singulární.

Pro výpočet lze v tomto případě využít pseudoinverze, konkrétně v MATLABu k tomuto slouží funkce *pinv()*.

```
1 Theta = pinv(X' * X) * X' * y;
```

Pokud $\lambda > 0$

$$\boldsymbol{\Theta} = \left(\boldsymbol{X}^\top \boldsymbol{X} + \lambda \underbrace{\begin{bmatrix} 0 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix}}_{\text{Matice musí být invertovatelná}}\right)^{-1} \boldsymbol{X}^\top y \tag{5.23}$$

5.4 Regularizace binární regrese

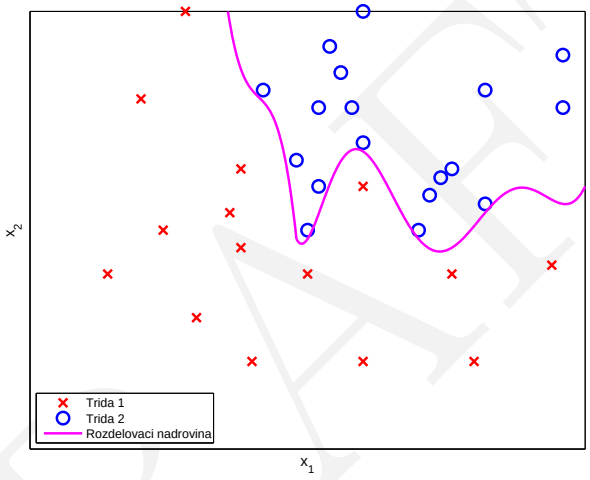
Pro binární regresi jsme dříve mluvili o dvou typech optimalizace. Mluvili jsme o tom, jak použít gradientní algoritmus a jak použít pokročilé metody optimalizace. V této sekci si ukážeme, jak tyto přístupy upravit, aby využily regularizace.

Regularizace binární regrese

Dříve jsme si ukázali, že binární regrese je náchylná na přetrénování v případě, kdy zvolíme vysoký stupeň polynomu reprezentující naší hypotézu, například

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = g\left(\vartheta_0 + \vartheta_1 x_1 + \vartheta_2 x_1^2 + \vartheta_3 x_1^2 x_2 + \vartheta_4 x_1^2 x_2^2 + \vartheta_5 x_1^2 x_2^3 + \ldots\right), \tag{5.24}$$

pro takovouto složitou hypotézu můžeme dostat dělící křivku velmi variabilní a velmi respektující rozdělení trénovacích dat, tedy křivku, která nedostatečně zobecňuje.



Obrázek 5.8: Možný výsledek s použitím hypotézy z rovnice 5.24.

Ztrátová funkce pro binární regresi má tvar

$$J(\boldsymbol{\Theta}) = -\left[\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right)\right) + \left(1 - y^{(i)}\right) \log\left(1 - h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right)\right)\right] \tag{5.25}$$

a její modifikací získáme regularizovaný tvar ztrátové funkce

$$J(\boldsymbol{\Theta}) = -\left[\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right)\right) + \left(1 - y^{(i)}\right) \log\left(1 - h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right)\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n} \vartheta_j^2 \tag{5.26}$$

přidaná suma je jen do n, jelikož se neregularizuje složka ϑ_0, která je vždy rovna jedné a není v součinu se žádnou proměnnou x, y. Tvar ztrátové funkce společně s regularizací bude naší rozdělovací nadrovinu více zobecňovat a pomůže nám řešit problém přetrénování.

Implementace bude provedena následovně: opakuj

$$\vartheta_0 = \vartheta_0 - \alpha \frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right) x_0^{(i)} \tag{5.27}$$

$$\vartheta_j = \vartheta_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right) x_j^{(i)} + \frac{\lambda}{m}\vartheta_j\right], \qquad j = 1, 2, 3, \ldots, n, \tag{5.28}$$

kde

$$h_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{\Theta}^\top \boldsymbol{x}}}. \tag{5.29}$$

Z rovnice 5.28 lze vidět, že platí

$$\frac{\partial}{\partial \vartheta_j} J(\boldsymbol{\Theta}) = \frac{1}{m} \sum_{i=1}^{m} \left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)} \right) x_j^{(i)} + \frac{\lambda}{m} \vartheta_j. \tag{5.30}$$

Regularizace pokročilé optimalizace

Nyní bude nastíněno jak využit pokročilé optimalizace v MATLABu. Nechť náš vektor parametrů $\boldsymbol{\Theta}$ odpovídá

$$\boldsymbol{\Theta} = [\vartheta_0, \vartheta_1, \ldots, \vartheta_n]^\top = [theta(1), theta(2), \ldots, theta(n+1)] \tag{5.31}$$

(z důvodu, že MATLAB indexuje od jedničky).

```
function [jVal, gradient] = costFunction(theta)
    jVal = [code to compute J(Theta)];
```

$$J(\boldsymbol{\Theta}) = \left[-\frac{1}{m} \sum_{i=1}^{m} y^{(i)} \log\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) \right) + \left(1 - y^{(i)}\right) \log\left(1 - h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right)\right) \right] + \frac{\lambda}{2m} \sum_{j=1}^{n} \vartheta_j^2 \tag{5.32}$$

dále výpočet gradientu

```
gradient(1) = [code to compute gradient(1)];
```

$$\frac{\partial}{\partial \vartheta_0} J(\boldsymbol{\Theta}) = \frac{1}{m} \sum_{i=1}^{m} \left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)} \right) x_0^{(i)}, \tag{5.33}$$

```
gradient(2) = [code to compute gradient(2)];
```

$$\frac{\partial}{\partial \vartheta_1} J(\boldsymbol{\Theta}) = \left[\frac{1}{m} \sum_{i=1}^{m} \left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)} \right) x_1^{(i)} \right] + \frac{\lambda}{m} \vartheta_1, \tag{5.34}$$

```
gradient(3) = [code to compute gradient(3)];
```

$$\frac{\partial}{\partial \vartheta_2} J(\boldsymbol{\Theta}) = \left[\frac{1}{m} \sum_{i=1}^{m} \left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)} \right) x_2^{(i)} \right] + \frac{\lambda}{m} \vartheta_2, \tag{5.35}$$

```
gradient(n+1) = [code to compute gradient(n+1)];
```

$$\ldots \tag{5.36}$$

Kapitola 6

Neuronové sítě

6.1 Nelineární hypotézy

6.2 Neurony a lidský mozek

6.3 Reprezentace modelu

6.4 Příklady

6.5 Klasifikace do více tříd

Kapitola 7

Neuronové sítě učení

7.1 Ztrátová funkce

7.2 Algoritmus zpětného šíření chyby

7.3 Gradient Checking

7.4 Náhodná inicializace

7.5 Příklady

Kapitola 8

Suport Vector Machine (SVM)

8.1 Cíl optimalizace

8.2 Large margin

8.3 Mathematic Behind Large Margin Classification

8.4 Kernels / Jádra

8.5 Kernel Trick

Bowl $x^2 + y^2$

8.6 Použití SVM

Kapitola 9

Shlukování

V této kapitole bude popsáno učení bez učitele, algoritmus K-Means, cíl optimalizace, náhodná inicializace a volba počtu shluků.

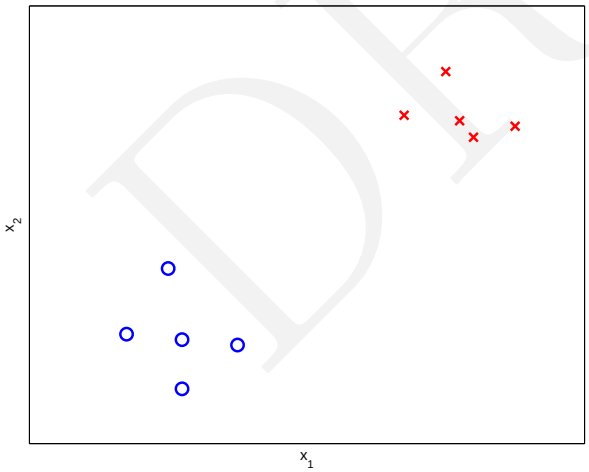
9.1 Učení bez učitele

Učení s učitelem

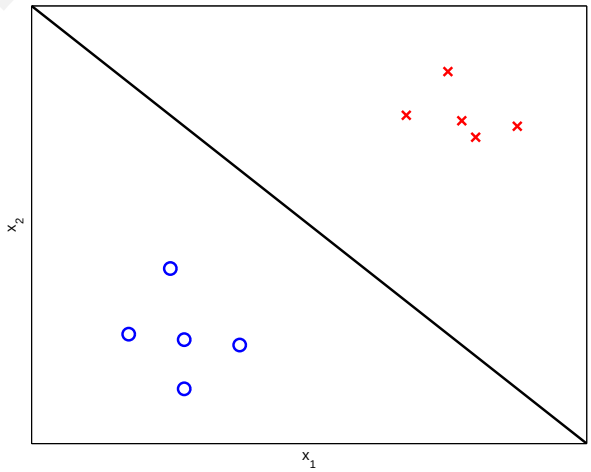
Typický problém učení s učitelem, kdy máme množinu vzorků $\boldsymbol{x}$ a k nim odpovídající informaci od učitele y. Trénovací množina má strukturu

$$\left\{\left(\boldsymbol{x}^{(1)}, y^{(1)}\right), \left(\boldsymbol{x}^{(2)}, y^{(2)}\right), \left(\boldsymbol{x}^{(3)}, y^{(3)}\right), \ldots, \left(\boldsymbol{x}^{(m)}, y^{(m)}\right)\right\}. \tag{9.1}$$

cílem je nalézt dělící nadrovinu, která bude nejlépe klasifikovat do dvou tříd.



Obrázek 9.1: Klasifikace do dvou tříd - učení s učitelem.



Obrázek 9.2: Klasifikace do dvou tříd s dělící nadrovinou - učení s učitelem.

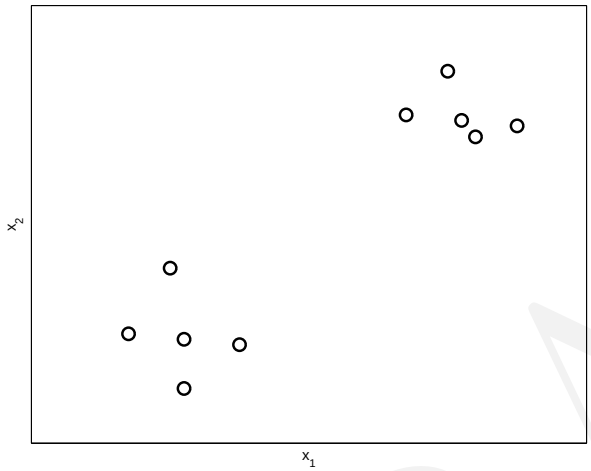
Na Obr. 9.1 a 9.2 barva a znak (kolečko nebo křížek) reprezentuje informaci od učitele.

Učení bez učitele

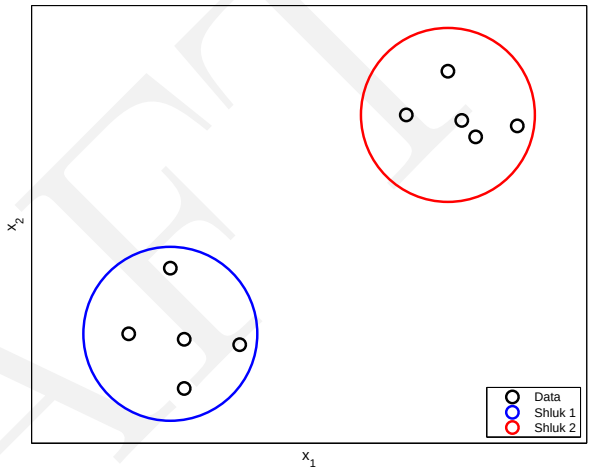
V případě učení bez učitele nemáme informaci od učitele do jaké třídy daný vzor patří. Trénovací množina má proto tvar

$$\left\{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}, \ldots, \boldsymbol{x}^{(m)}\right\}, \tag{9.2}$$

cílem učícího se algoritmu je nalézt strukturu mezi neznámými daty a klasifikovat jednotlivé vzory do odpovídajících tříd. Je zřejmé, že v případě učení bez učitele postrádáme důležitou informaci o počtu tříd. V extrémních případech může být počet tříd roven jedné a nebo počtu trénovacích vzorů. Na Obr. 9.3 a 9.4 lze vidět, že informace od učitele ve formě barvy a znaku chybí.



Obrázek 9.3: Klasifikace do dvou tříd - učení bez učitele.



Obrázek 9.4: Klasifikace do dvou tříd s naznačenými shluky - učení bez učitele.

Příklad klasifikace v případě učení bez učitele je **shlukování** , kde se jednotlivé třídy nazývají **shluky**.

9.2 K-Means

9.3 Cíl optimalizace

9.4 Náhodná inicializace

9.5 Volba počtu shluků

Kapitola 10

Redukce dimenze

10.1 Analýza hlavních komponent

Principal Component Analysis, PCA

10.2 Linear discriminant analysis

Linear discriminant analysis, LDA

Kapitola 11

Ostatní

11.1 GMM

11.2 Jak postavit test

testovaci trenovaci a cross validation sada

11.3 Vyhodnocování úspěšnosti klasifikace

recall, precision a F1 skore