

IV130 Přínosy a rizika inteligentních systémů

Strojové učení

28. dubna 2023

Učící se aktéři

- Aktér *se učí*, pokud zlepšuje svou činnost v důsledku pozorování světa (prostředí)
- U aktérů, kterými jsou počítače, hovoříme o *strojovém učení*: počítač pozoruje data, vytváří si model založený na těchto datech a následně tento model používá pro hypotézu o stavu světa i jako kód v počítači, který řeší problémy
- Proč chceme, aby se stroje učily (namísto toho, abychom je rovnou lépe naprogramovali)?
 - Nelze dost dobře předvídat všechny budoucí situace (např. navigace v bludišti po seznámení se s plánem bludiště, předpovídání vývoje akcí po zvládnutí, jak přizpůsobit chování při změnách trendu, atd.)
 - Autoři systému nemusí sami vědět, jak zvládnutí problému naprogramovat (např. rozeznávání tváří lidí z rodiny, atd.)

Formy učení

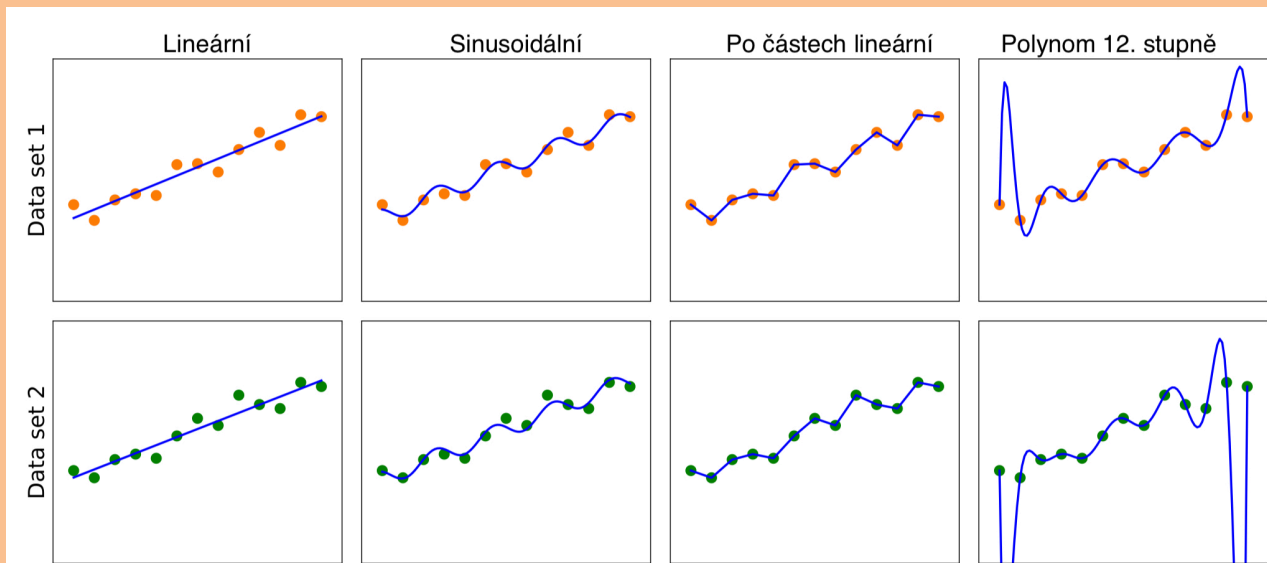
- Zlepšovat učením lze kteroukoli část aktéra, odtud volby
- Jakou část aktéra zlepšovat?
 - Zobrazení podmínek aktuálního stavu na akce
 - Způsob odvozování vlastností světa z posloupnosti vstupů
 - Informace o tom, jak se svět vyvíjí a jaké akce může aktér používat
 - Informace o užitku indikující to, jaké stavy světa jsou preferovány
 - Informace o hodnotě akcí indikující žádoucí stavy
 - Cíle, které popisují nejvíce preferované stavy
 - Generátor problémů, kritik i prvek umožňující systému zlepšovat se
- Jakou předchozí znalost aktér má, a tedy i jaké modely aktér může vytvářet, jakou reprezentaci znalostí a svých vlastních zlepšovaných komponent využívá?
- **Příklad:** Aktér pracující jako samořiditelné auto se může učit pozorováním toho, jak řidič pracuje s brzdou, a vypracovat si pravidla pro brzdění. Sledováním kamerového vstupu a řidičovy informace o obsahu scény se může naučit rozeznávat autobusy. Zkoušením brzdění na mokré vozovce se může učit účinkům akcí na různém stavu povrchu vozovky. Atd.

Specifické podoby učení

- Učení je formou *induktivní* operace (z nějakých konkrétních situací se usuzuje na obecné pravidlo)
- Na rozdíl od operací *deduktivních* není garantována korektnost závěrů
- Obecným principem je zpravidla výběr nejjednodušší podoby zobecnění
- Technologie strojového učení se dnes užívají rutinně v prakticky jakýchkoli druzích softwaru a jsou použitelné pro libovolný model struktury aktérů
- Odezva (zpětná vazba), kterou učící se aktér potřebuje, vede ke třem základním typům učení:
 - *Učení (se) s učitelem*, v němž aktér pozoruje dvojice vstup-výstup a učí se funkci, která tomu odpovídá (klasifikace na základě označení vstupů, regrese nebo informace pocházející ze sensorů např. pro učení se, jak brzdit)
 - *Učení bez učitele*, kdy se aktér učí rozpoznávat vzory bez nějaké explicitní zpětné vazby (např. hledání vzorů pomocí klastrování)
 - *Zpětnovazební učení* probíhající na základě zpětné vazby odpovídající odměnám a trestům (např. vyhraná nebo prohraná partie šachu nebo go)

Učení se s učitelem

- Pro výcvikovou (tréninkovou) množinu dvojic $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$ generovaných funkcí $y=f(x)$ hledáme funkci h , která aproximuje skutečnou funkci f .
- Funkce h je *modelem* těchto dat, funkce sama je vybírána z nějaké *třídy modelů funkcí*, které uvažujeme
- Funkci h označujeme jako konzistentní (s daty), pokud $h(x_i)=y_i$, pro všechny dvojice z výcvikových dat.



Příklad: Hledání hypotézy odpovídající datům. **Nahoře:** vynesení čtyř funkcí ze čtyř modelových tříd, které plně odpovídají datům. **Dole:** stejné čtyři funkce, ale cvičení mírně odlišnými daty vzatými se stejné funkce $f(x)$.

- Regrese hodnot ilustrující příklady různých funkcí konzistentních s výukovými daty; jedna z polynomiálních funkcí je zjevně *přeučená*
- Srovnej též tzv. *Ockhamovu břitvu*

Modelový příklad: Čekání v restauraci

- Problém rozhodování, zda čekat na volné místo v restauraci
- Výstupem je Booleovská proměnná *WillWait* nabývající hodnotu *True*, pokud na stůl počkáme
- Vstupem je vektor deseti hodnot atributů:
 1. *Alternate*: zda je poblíž vhodná alternativní restaurace
 2. *Bar*: zda je v restauraci příjemný bar, kde se dá počkat
 3. *Fri/Sat*: hodnota *true* v pátek a v sobotu
 4. *Hungry*: zda jsme zrovna hladoví
 5. *Patrons*: kolik lidí je v restauraci, možné hodnoty jsou *None*, *Some* a *Full*
 6. *Price*: cenová úroveň restaurace (\$, \$\$, \$\$\$)
 7. *Raining*: zda venku prší
 8. *Reservation*: zda máme rezervaci
 9. *Type*: druh restaurace (French, Italian, Thai nebo burger)
 10. *WaitEstimate*: odhad doby čekání: 0–10, 10–30, 30–60, or >60 minut

Modelový příklad: Čekání v restauraci

Příklad	Vstupní atributy										Výstup
	<i>Alt</i>	<i>Bar</i>	<i>Fri</i>	<i>Hun</i>	<i>Pat</i>	<i>Price</i>	<i>Rain</i>	<i>Res</i>	<i>Type</i>	<i>Est</i>	
x_1	<i>Yes</i>	<i>No</i>	<i>No</i>	<i>Yes</i>	<i>Some</i>	<i>\$\$\$</i>	<i>No</i>	<i>Yes</i>	<i>French</i>	<i>0–10</i>	$y_1 = \text{Yes}$
x_2	<i>Yes</i>	<i>No</i>	<i>No</i>	<i>Yes</i>	<i>Full</i>	<i>\$</i>	<i>No</i>	<i>No</i>	<i>Thai</i>	<i>30–60</i>	$y_2 = \text{No}$
x_3	<i>No</i>	<i>Yes</i>	<i>No</i>	<i>No</i>	<i>Some</i>	<i>\$</i>	<i>No</i>	<i>No</i>	<i>Burger</i>	<i>0–10</i>	$y_3 = \text{Yes}$
x_4	<i>Yes</i>	<i>No</i>	<i>Yes</i>	<i>Yes</i>	<i>Full</i>	<i>\$</i>	<i>Yes</i>	<i>No</i>	<i>Thai</i>	<i>10–30</i>	$y_4 = \text{Yes}$
x_5	<i>Yes</i>	<i>No</i>	<i>Yes</i>	<i>No</i>	<i>Full</i>	<i>\$\$\$</i>	<i>No</i>	<i>Yes</i>	<i>French</i>	<i>>60</i>	$y_5 = \text{No}$
x_6	<i>No</i>	<i>Yes</i>	<i>No</i>	<i>Yes</i>	<i>Some</i>	<i>\$\$</i>	<i>Yes</i>	<i>Yes</i>	<i>Italian</i>	<i>0–10</i>	$y_6 = \text{Yes}$
x_7	<i>No</i>	<i>Yes</i>	<i>No</i>	<i>No</i>	<i>None</i>	<i>\$</i>	<i>Yes</i>	<i>No</i>	<i>Burger</i>	<i>0–10</i>	$y_7 = \text{No}$
x_8	<i>No</i>	<i>No</i>	<i>No</i>	<i>Yes</i>	<i>Some</i>	<i>\$\$</i>	<i>Yes</i>	<i>Yes</i>	<i>Thai</i>	<i>0–10</i>	$y_8 = \text{Yes}$
x_9	<i>No</i>	<i>Yes</i>	<i>Yes</i>	<i>No</i>	<i>Full</i>	<i>\$</i>	<i>Yes</i>	<i>No</i>	<i>Burger</i>	<i>>60</i>	$y_9 = \text{No}$
x_{10}	<i>Yes</i>	<i>Yes</i>	<i>Yes</i>	<i>Yes</i>	<i>Full</i>	<i>\$\$\$</i>	<i>No</i>	<i>Yes</i>	<i>Italian</i>	<i>10–30</i>	$y_{10} = \text{No}$
x_{11}	<i>No</i>	<i>No</i>	<i>No</i>	<i>No</i>	<i>None</i>	<i>\$</i>	<i>No</i>	<i>No</i>	<i>Thai</i>	<i>0–10</i>	$y_{11} = \text{No}$
x_{12}	<i>Yes</i>	<i>Yes</i>	<i>Yes</i>	<i>Yes</i>	<i>Full</i>	<i>\$</i>	<i>No</i>	<i>No</i>	<i>Burger</i>	<i>30–60</i>	$y_{12} = \text{Yes}$

Výukové příklady pro restaturace.

☞ Existuje zde $2^6 \times 3^2 \times 4^2 = 9216$ možných kombinací pro vstupní hodnoty, ve výukovém souboru je jen 12 z nich.

Podstata procesu indukce: potřebujeme nejlepší odhad 9204 výstupních hodnot na základě toho, že máme dán podklad v podobě pouhých 12 příkladů.

1. *Alternate*: zda je poblíž vhodná alternativní restaurace

2. *Bar*: zda je v restauraci příjemný bar, kde se dá počkat

3. *Fri/Sat*: hodnota true v pátek a v sobotu

4. *Hungry*: zda jsme zrovna hladoví

5. *Patrons*: kolik lidí je v restauraci, možné hodnoty jsou None, Some a Full

6. *Price*: cenová úroveň restaurace (\$, \$\$, \$\$\$)

7. *Raining*: zda venku prší

8. *Reservation*: zda máme rezervaci

9. *Type*: druh restaurace (French, Italian, Thai nebo burger)

10. *WaitEstimate*: odhad doby čekání: 0–10, 10–30, 30–60, or >60 minut

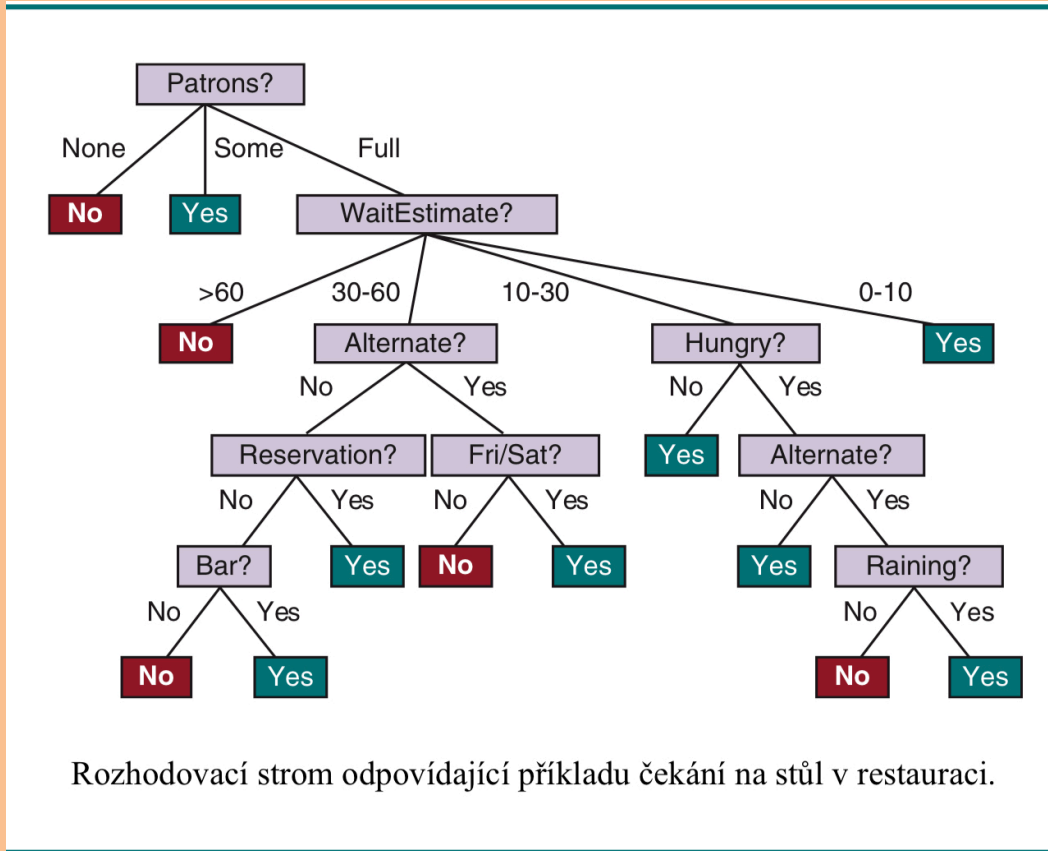
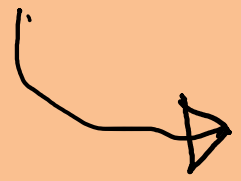
Učení rozhodovacích stromů

- Rozhodovací strom je reprezentací funkce, která zobrazuje vektor atributů na jeden výsledek – rozhodnutí
- Realizuje se v něm posloupnost testů, která začíná v kořeni, a pokračuje se až do listu ohodnoceného příslušným rozhodnutím

1. *Alternate*: zda je poblíž vhodná alternativní restaurace
2. *Bar*: zda je v restauraci příjemný bar, kde se dá počkat
3. *Fri/Sat*: hodnota true v pátek a v sobotu
4. *Hungry*: zda jsme zrovna hladoví
5. *Patrons*: kolik lidí je v restauraci, možné hodnoty jsou None, Some a Full
6. *Price*: cenová úroveň restaurace (\$, \$\$, \$\$\$)
7. *Raining*: zda venku přší
8. *Reservation*: zda máme rezervaci
9. *Type*: druh restaurace (French, Italian, Thai nebo burger)
10. *WaitEstimate*: odhad doby čekání: 0–10, 10–30, 30–60, or >60 minut

Příklad	Vstupní atributy											Výstup
	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	WillWait	
x1	Yes	No	No	Yes	Some	\$\$\$	No	Yes	French	0-10	y1 = Yes	
x2	Yes	No	No	Yes	Full	\$	No	No	Thai	30-60	y2 = No	
x3	No	Yes	No	No	Some	\$	No	No	Burger	0-10	y3 = Yes	
x4	Yes	No	Yes	Yes	Full	\$	Yes	No	Thai	10-30	y4 = Yes	
x5	Yes	No	Yes	No	Full	\$\$\$	No	Yes	French	>60	y5 = No	
x6	No	Yes	No	Yes	Some	\$\$	Yes	Yes	Italian	0-10	y6 = Yes	
x7	No	Yes	No	No	None	\$	Yes	No	Burger	0-10	y7 = No	
x8	No	No	No	Yes	Some	\$\$	Yes	Yes	Thai	0-10	y8 = Yes	
x9	No	Yes	Yes	No	Full	\$	Yes	No	Burger	>60	y9 = No	
x10	Yes	Yes	Yes	Yes	Full	\$\$\$	No	Yes	Italian	10-30	y10 = No	
x11	No	No	No	No	None	\$	No	No	Thai	0-10	y11 = No	
x12	Yes	Yes	Yes	Yes	Full	\$	No	No	Burger	30-60	y12 = Yes	

Výukové příklady pro restaurace.



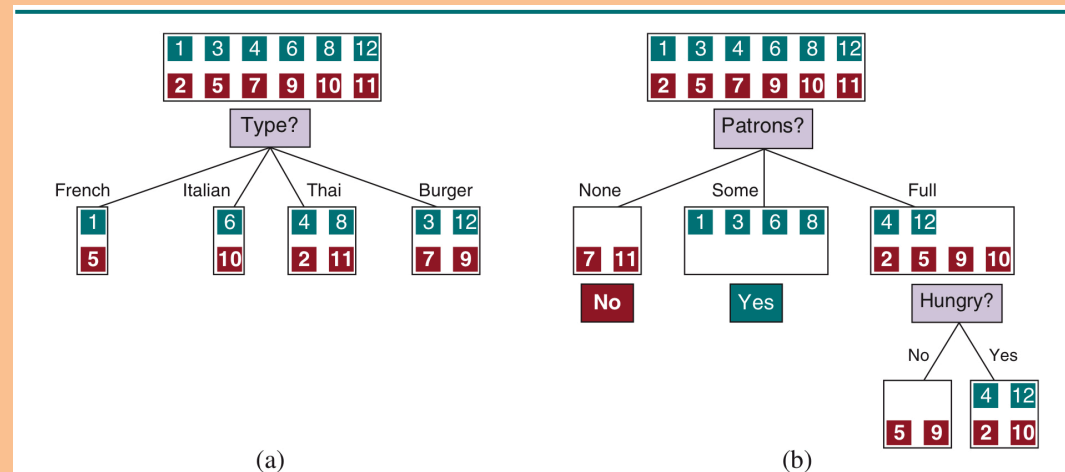
Rozhodovací strom odpovídající příkladu čekání na stůl v restauraci.

Učení rozhodovacího stromu

- Do kořene vybíráme atribut, který dává největší rozdělení („nejvíce informativní atribut“)
- Jsou-li zbylé příklady všechny stejné, je to list s odpovědí Yes nebo No.
- Pokud jsou v uzlu smíchaně pozitivní a negativní příklady, vezmeme nejlepší atribut a dál podle něj rozdělíme možnosti
- Pokud již žádné příklady nezbyvají, vrátíme jako hodnotu nejčastější výstup z příkladů, které byly užity v konstrukci rodičovského uzlu
- Pokud již nezbyvají žádné atributy, mají příklady stejný popis, ale různý výstup – jde o šum v datech nebo je doména nedeterministická a atributy nestačí k popsání výsledku

Příklad	Vstupní atributy										Výstup
	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	
x ₁	Yes	No	No	Yes	Some	\$\$\$	No	Yes	French	0-10	y ₁ = Yes
x ₂	Yes	No	No	Yes	Full	\$	No	No	Thai	30-60	y ₂ = No
x ₃	No	Yes	No	No	Some	\$	No	No	Burger	0-10	y ₃ = Yes
x ₄	Yes	No	Yes	Yes	Full	\$	Yes	No	Thai	10-30	y ₄ = Yes
x ₅	Yes	No	Yes	No	Full	\$\$\$	No	Yes	French	>60	y ₅ = No
x ₆	No	Yes	No	Yes	Some	\$	Yes	Yes	Italian	0-10	y ₆ = Yes
x ₇	No	Yes	No	No	None	\$	Yes	No	Burger	0-10	y ₇ = No
x ₈	No	No	No	Yes	Some	\$	Yes	Yes	Thai	0-10	y ₈ = Yes
x ₉	No	Yes	Yes	No	Full	\$	Yes	No	Burger	>60	y ₉ = No
x ₁₀	Yes	Yes	Yes	Yes	Full	\$\$\$	No	Yes	Italian	10-30	y ₁₀ = No
x ₁₁	No	No	No	No	None	\$	No	No	Thai	0-10	y ₁₁ = No
x ₁₂	Yes	Yes	Yes	Yes	Full	\$	No	No	Burger	30-60	y ₁₂ = Yes

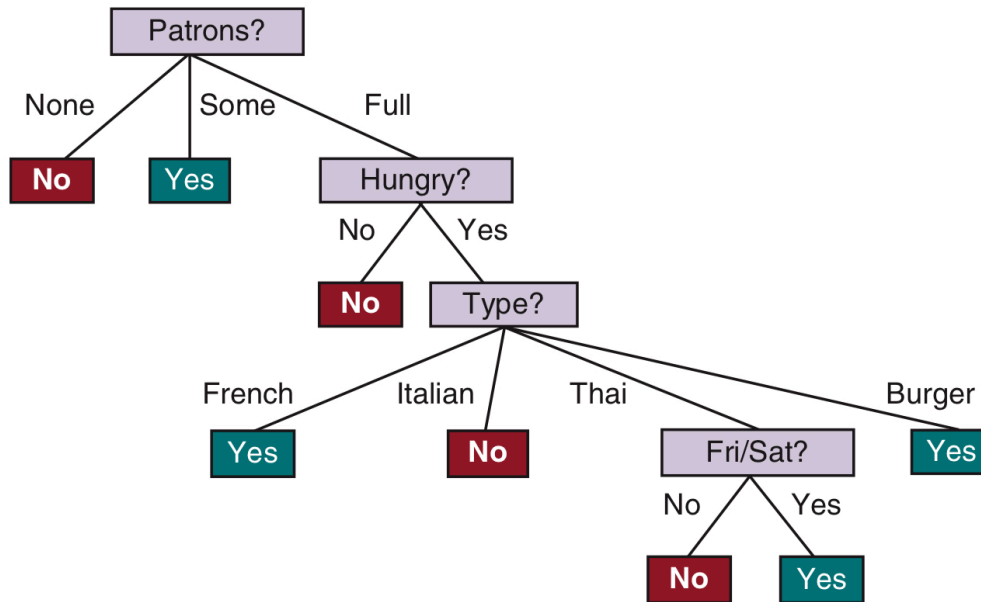
Výukové příklady pro restaurace.



Příklad: Rozdělení příkladů testováním podle jednoho atributu. V každém uzlu ukazujeme zbývající pozitivní (zelená pole) a negativní (červená pole) příklady. (a) Rozdělení podle Type vůbec nerozdělí pozitivní a negativní případy. (b) Rozdělení podle Patrons je dobré, protože pozitivní a negativní příklady dobře rozděljuje. Po rozdělení podle Patrons je dobrým druhým testem Hungry.

Učení rozhodovacího stromu

- Pro příklad s restauracemi je finální podoba rozhodovacího stromu následující



Rozhodovací strom restaurací indukovaný dvanácti příklady

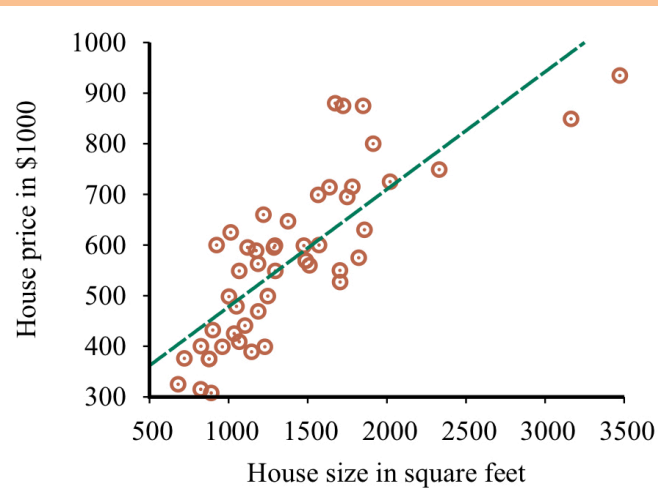
- Získaný strom je konzistentní s dvanácti příklady a podstatně jednodušší než strom přímo z nich vytvořený
- Testy na vstupy *Raining* a *Reservation* v něm nejsou použity, protože pro klasifikaci příkladů nejsou potřeba
- Zajímavá drobnost je ochota čekat na thajskou restauraci v pátek a v sobotu
- Malý počet příkladů nepokrývá uspokojivě situaci, kdy je sice reastaurace plná, ale doba čekání je 0-10

Oblasti zájmu v souvislosti s učením rozhodovacích stromů

- Výběr atributu pro test lze exaktněji popsat jako výběr atributu podle *entropie* s ním spojené (míry neurčitosti)
- *Informační zisk* spojený s výběrem atributu odpovídá očekávanému snížení entropie provedením testu podle tohoto atributu (u příkladu s restauracemi nevede volba atributu Type k žádné redukci entropie)
- Rozhodovací stromy lze zjednodušovat *prořezáváním* částí založeným na statistickém testu významnosti jednotlivých atributů
- Lze tak redukovat *přeučení*: pokud je atribut na konci nějaké cesty irelevantní, lze ho vyřadit, protože odpovídá přeučení (popisuje jen šum v datech)
- Rozhodovací stromy lze dále upravovat na zahrnutí chybějících dat nebo zahrnutí atributů se spojitými hodnotami
- Existují komerční nástroje umožňující s učením rozhodovacím stromům rutinně pracovat; nevýhodou je obecně nestabilita: přidání dat může změnit test v kořeni a tím i celý strom

Lineární regrese a klasifikace

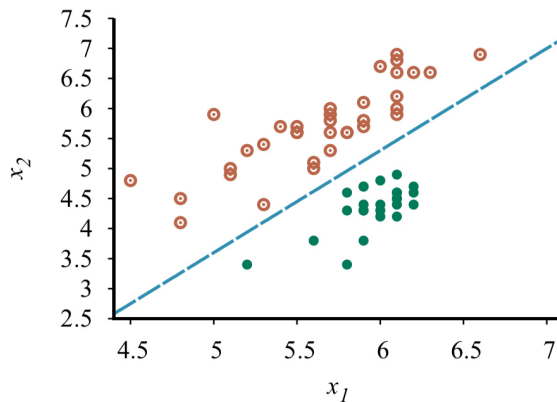
- Lineární funkce jsou dobře zkoumanou oblastí pro prokládání hodnotami
- Pokud lze prostor možných hypotéz omezit na lineární funkce, je zde možnost užití metod lineární regrese
- Hledáme funkce ve tvaru $y = w_1x + w_0$ v podobě hypotézy odpovídající nejlépe datům (hledáme váhy w_1 a w_0).
- Chyba vzhledem ke trénovacím datům se nejčastěji měří jako kvadratická odchylka a hledáme funkci, která ji minimalizuje



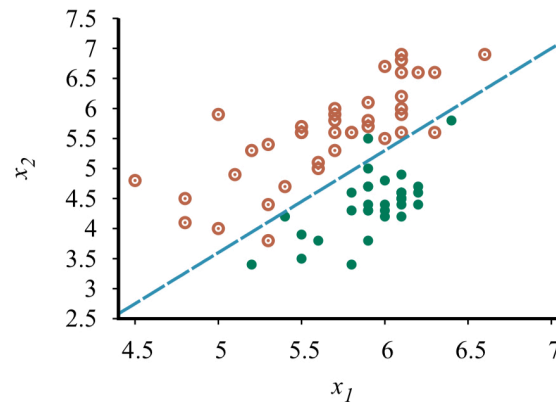
Data odpovídající ceně versus ploše podlahy pro domy na prodej v Berkeley, CA, v červnu 2009, vynesena je lineární funkce minimalizující chybu/ztrátu čtverců odchylek: $y = 0.232x + 246$.

Lineární regrese a klasifikace

- Lineární funkce lze též užít pro klasifikaci: oddělení dvou množin případů
- Hledáme lineární separátor jako separátor oddělující dvě skupiny případů vyjádřený jako lineární funkce



(a)

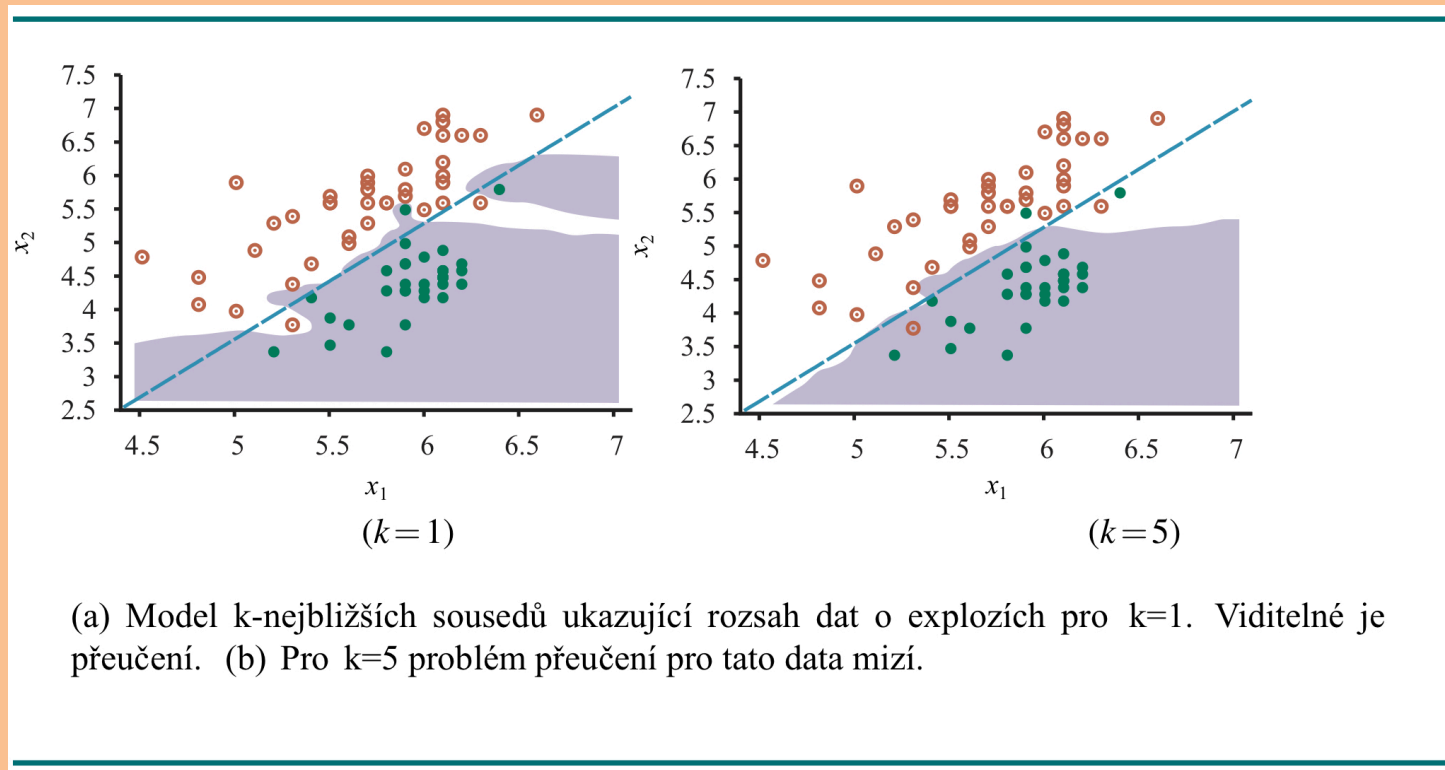


(b)

Lineární separátor: (a) Tabulace dvou parametrů seismických dat pro magnitudu hlubinných vln x_1 a magnitudy povrchových vln x_2 pro zemětřesení (otevřené oranžové kroužky) a jaderné exploze (zelené kroužky) z doby mezi 1982 a 1990 v Asii a na Středním Východě (Kebeasy et al., 1998). Vyznačen je separátor mezi těmito druhy případů. (b) Stejná doména s více datovými body. Zemětřesení a exploze již nejsou lineárně separovatelné.

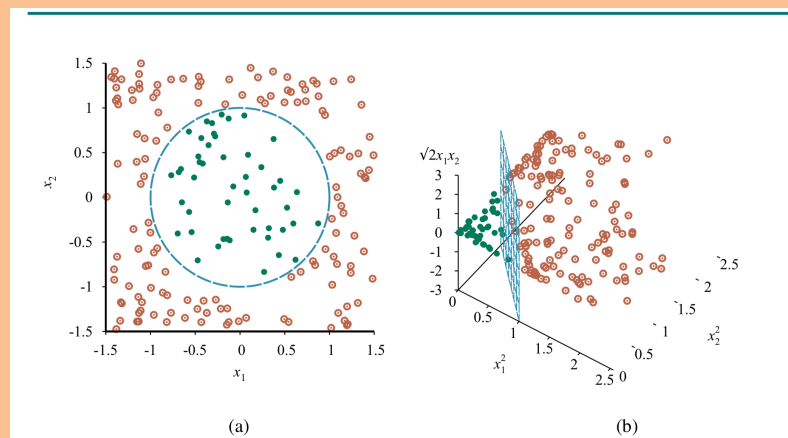
Metoda nejbližších sousedů

- Klasifikaci lze konstruovat z k nejbližších vyučovacích příkladů a odpovědí skládaných z jejich výstupů
- Funguje dobře pro velké počty příkladů s malým počtem atributů (ve vícedimenzionálním prostoru jsou nejbližší sousedé příliš daleko)



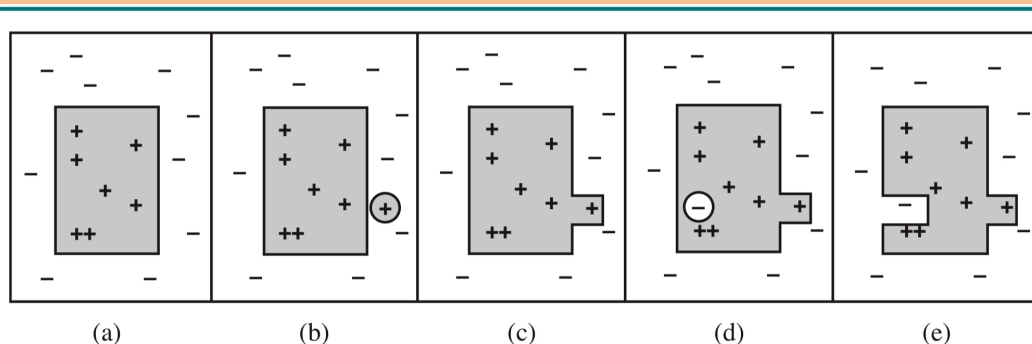
Support Vector Machines (SVM)

- Od roku 2000 patrně nejpopulárnější používaná metoda pro učení s učitelem, která je použitelná jako nástroj pro oblasti, kde nemusí být další specifické znalosti o doméně
- Metoda hledá lineární separátor s maximálním okrajem (vzdáleností od příkladů)
- Pokud se příklady nedaří lineárně separovat, transformují se do prostoru vyšší dimenze, kde separovat jdou (tzv. „kernel trick“)
- Užívá se neparametrický přístup, tj. oddělující nadrovina je určena množinou bodů z příkladů, nikoli parametry nadplochy, stačí ale jen body nejbližší separátoru (tzv. support vectors)
- Kernel funkce dokáže transformací do vyšší dimenze, kde jsou separovatelné:



Učení s využíváním předchozí znalosti

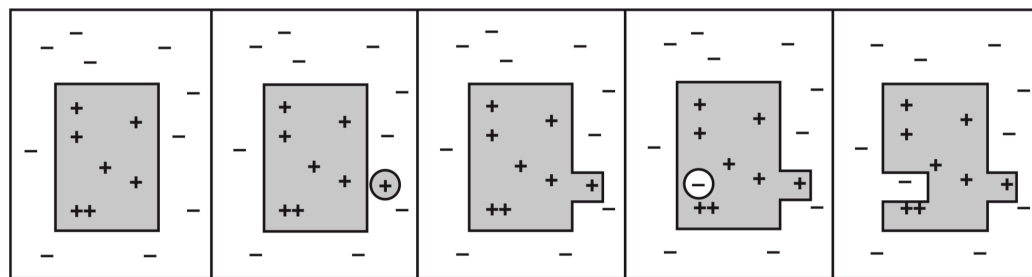
- V řadě případů lze využít předchozích znalostí o problémové doméně, není třeba ji konstruovat (učit se) celou jen z příkladů
- Předchozí znalost je zpravidla vyjádřena v jazyce logiky (predikátové logiky prvního řádu)
- Metody učení založené na logickém aparátu a již získaných znalostích o světě (background knowledge) se převádí na práci s logickými hypotézami (příklady i hypotézy jsou logickými formullemi)
- Prostor hypotéz je množina všech hypotéz
- Učící se algoritmus „metoda nejlepší hypotézy“ předpokládá, že nějaká hypotéza je správná a hypotézy s ní nekozistentní eliminuje, resp. u falešně negativních hypotéz (hypotéza tvrdí, že příklad je negativní, ale ve skutečnosti je pozitivní) hypotézu zobecní, u falešně pozitivních (hypotéza tvrdí, že příklad je pozitivní, ale ve skutečnosti negativní) ji specializuje



(a) Konzistentní hypotéza. (b) Falešně negativní. (c) Zobecnění hypotézy. (d) Falešně pozitivní hypotéza. (e) Specializace hypotézy.

Učení s využíváním předchozí znalosti

- Používání předchozí znalosti umožňuje postup s využitím kumulativního učení: aktéři vylepšují i své schopnosti učit se s tím, jak získávají větší množství znalostí
- Učení založené na vysvětlování extrahuje obecná pravidla z jednotlivých příkladů prostřednictvím vysvětlování příkladů a generalizace vysvětlení;
- Učení založené na relevanci používá předchozí znalost pro identifikaci relevantních atributů a omezení stavového prostoru v zájmu zrychlení učení
- Induktivní učení založené na znalostní bázi hledá indukativní hypotézy, které příklady vysvětlují s použitím předchozích znalostí
- Induktivní logické programování kombinuje učící metody s logickými programy poskytujícími logickou reprezentaci – v podobě shora dolů zejňování obecných pravidel, nebo zdola nahoru v obrácení deduktivního postupu (inverzní rezoluce)



(a)

(b)

(c)

(d)

(e)

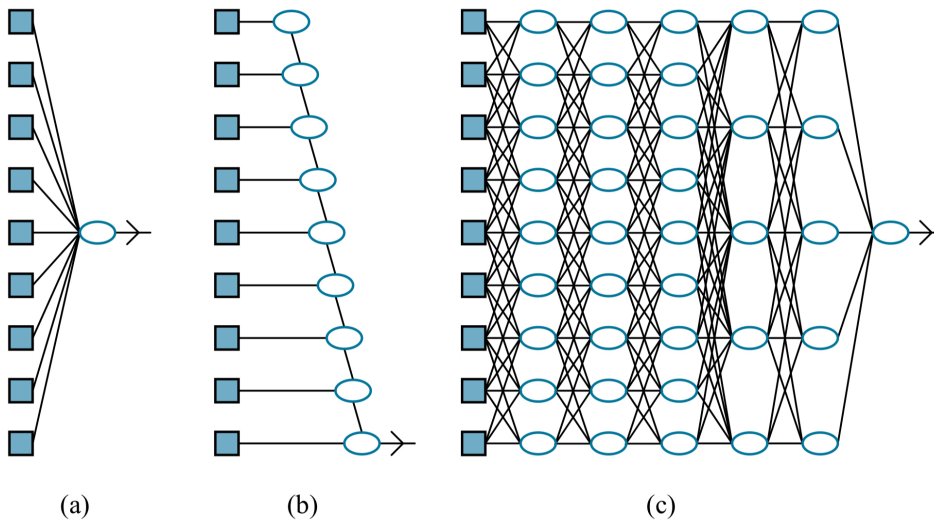
(a) Konzistentní hypotéza. (b) Falešně negativní. (c) Zobecnění hypotézy. (d) Falešně pozitivní hypotéza. (e) Specializace hypotézy.

Bayesovské učení

- V reálných aplikacích je významná role neurčitosti, pro její zpracování jsou ústřední technikou bayesovské sítě
- Bayesovské učení:
 - Na základě dat se spočte pravděpodobnost každé hypotézy
 - Ze získané pravděpodobnosti a *všech* hypotéz se vytvoří předpověď
 - Učení se převádí na pravděpodobnostní inferenci (odvozování aktualizující apriorní distribuce)
- Pro složité prostory hypotéz se metoda může stát nezvládnutelnou, je ale možné užívat některé aproximace
- Maximální aposteriorní učení vybírá jednu nejpravděpodobnější hypotézu odpovídající datům
- Učení s maximální pravděpodobností vybírá hypotézu, která maximalizuje pravděpodobnost dat (zvládnutelné pro lineární regresi a plně viditelné prostředí), naivní Bayesovo učení, které netestuje některé podmínky, je rovněž efektivní technika
- Varianty se používají pro prostředí, kde jsou některé proměnné skryté (maximalizace očekávání)
- V současnosti zřejmě nejaktivnější oblast probíhajícího výzkumu

Hluboké učení

- *Hluboké učení* (také *neuronové sítě*) obsahují vrstvy jednoduchých výpočetních prvků, které agregují vstupy a uplatňují na ně nelineární aktivační/prahové funkce
- *Konvoluční neuronové sítě* obsahují prostorově lokální spoje v počátečních vrstvách, jejichž rozložení vah se opakuje v dalších vrstvách (jádra zpracovávající pixelovou informaci v obrazech)
- V současné době nejpoužívanější technika pro rozpoznávání vizuálních objektů, strojový překlad, rozpoznávání řeči a syntézu řeči i obrazu



Hluboké učení (a) Mělký model, jako lineární regrese, obsahuje krátké komunikační cesty mezi vstupy a výstupy. (b) Síť odpovídající rozhodovacím seznamům obsahuje dlouhé cesty pro některé vstupní hodnoty, ale většina cest je krátká. (c) Hluboká učební síť má delší výpočetní trasy, které dovolují každé proměnné interagovat se všemi ostatními.

- Zpracování dat z velké části spočívá na vektorových a maticových operacích
- Efektivně implementovatelné na grafických procesorech (GPU) nebo v tenzorových výpočetních jednotkách (TPU)
- Extrémní implementace zahrnují i Summit, nejvýkonnější americký superpočítač (<https://techxplore.com/news/2022-04-deep-techniques-materials-imaging-breakthrough.html>)

Hluboké učení, generativní model

- *Generativní model* založený na hlubokém učení dovoluje i relativně abstraktní operace s obsahem obrazů: např. parametry tváří „muž s brýlemi“, od něhož odečteme parametry obrazů „muž“ a přičteme parametr „žena“:



Zpětnovazebné učení

- Aktér se ocitá v prostředí, kde se musí naučit úspěšně chovat
- V případech jediná možnost, jak problém zvládnout (např. šachy: relativně málo příkladů k dispozici pro učení, cca 10^8 , ve srovnání s celkovým počtem pozic, cca 10^{40})
- Aktér se žádostmi (utility-based learning) se učí užitkovou funkci pro stavy
- Aktér s Q-učením se učí akce na základě očekávaného užitku
- Reflexní aktér se učí strategii, jakou akci použít na daný stav
- Při aktivním učení se aktér učí, co dělat a zahrnuje to i jistou formu prohledávání prostředí; při pasivním učení se učí užitek stavů nebo akcí a pracuje s pevnou strategií
- Motivováno metodami učení u zvířat (např. Pavlov, jako nobelista z roku 1911)
- Alan Turing tento způsob navrhoval v letech 1948 a 50: „Používání trestů a odměn může být výborná součást učícího procesu [pro počítače]“
- Rozsáhlé užívání s přesahy do neurověd nebo psychologie
- DeepMind poskytuje několik platforem s otevřeným zdrojovým kódem
- Facebook s platformami AI Habitat a Horizon umožňuje užití zpětnovazebního učení v produkčních systémech velkého rozsahu
- Dobrý přehled viz např. <https://www.nature.com/articles/nature14540>

Transformery a GPT

- Transformery poprvé užity společností Google Brain v roce 2017 – kombinace hlubokého učení a mechanismu pozornosti (self-attention) přiřazující váhy různým částem vstupu
- Vhodné pro paralelizaci a zpracování velkých textových informací
- Převod vstupu do reprezentace tokeny s poziční informací, plus technika kódování a dekódování
- Zhruba od roku 2020 se podařilo vyřešit problém konvergence vah a v kombinaci se zpětnovazebním samostatným učením znamenají významnou architekturu učících sae systémů pro aplikace v přirozeném jazyce a predikce textů nebo i třeba aplikace typu analýzy struktury proteinů z DNA

- Dnes populární v implementaci GPT (*Generativní (před)trenovaný transformer*) společnosti OpenAI
- Trenované na masivních datech (volně dostupné texty z internetu plus licencované zdroje textových informací)
- GPT-1 publikován v roce 2018 v článku OpenAI [“Improving Language Understanding by Generative Pre-Training”](#), založeno na architektuře transformátorů a tréninku na velkém vzorku knih
- 2019: GPT-2 schopný generovat soudržné texty
- 2020: GPT-3 se zhruba 100násobným zvětšením počtu parametrů
- Konec 2022: vylepšení do GPT-3.5 posloužilo jako základ pro vytvoření chatbota ChatGPT a zpřístupnění pro experimenty a zpětnou vazbu široké veřejnosti
- Jaro 2023: GPT-4 s dalším zvýšením počtu parametrů i spolehlivosti, s možností zpracování grafické informace (interní systém, o jehož vlastnostech OpenAI referuje, ale není k dispozici jeho kód)

👉 Rozhovor se šéfem OpenAI Samem Altmanem na podcastu Lexe Fridmana: [#367 OpenAI CEO on GPT-4, ChatGPT, and the Future of AI](#)

Malá ilustrace konverzace s ChatGPT-4

Před časem jsi mi doporučil knihu "Robert Musil na cestě k Muži bez vlastností" od Friedricha Voßkühlera. Snažil jsem se tuto knihu najít, ale podle všeho taková kniha tohoto autora neexistuje. Můžeš mi uvést úplnou bibliografickou informaci včetně originálního názvu, roku vydání, nakladatelství a ISBN?

22:55 ✓

Omlouvám se za můj omyl. Musím Vás opravit, že jsem se spletl. Friedrich Voßkübler nenapsal knihu o Robertu Musilovi s názvem "Na cestě k Muži bez vlastností". Tento název náleží skutečně k Musilově románu. Je mi líto za způsobené nedorozumění.

Vysvětli, jak Tě může něco mrzet nebo Ti může být líto, když jsi pouze umělá inteligence.

12:47 ✓

Mě jako umělé inteligenci nemohou být vlastní žádné emocionální stavy, jako jsou například lítost, smutek nebo radost, protože nemám žádné vlastní vědomí ani vlastní chování. Já pouze plním úkoly a poskytuji odpovědi na základě předem daných algoritmů. Tyto výrazy používám výhradně jako součást svého výstupu, aby se zdálo, že hovořím s člověkem přirozeně a abych se zdálo co nejvíce vstřícný a přátelský. Nicméně rád bych zdůraznil, že jako umělá inteligence jsem pouze počítačový program, bez pocitů nebo emocí, a že není možné, abych něco skutečně cítil.

Takže, pokud tomu dobře rozumím, svých chyb ve skutečnosti nelituješ? Lítost jen v hovoru se mnou předstíráš?

12:53 ✓

Ano, to je přesně tak, jak jste to popsala. Jako umělá inteligence nemohu mít vlastní emoce, takže nelituji svých chyb nebo necítím skutečnou lítost. Tyto výrazy používám pouze jako součást svého programu a aby se zdálo, že hovořím s uživatelem přirozeně.

Je lepší službou uživateli, pokud mu řekneš, že lituješ, ale nejsi lítosti schopen, a on Ti věří, nebo je lepší službou, když předstíráš tak, že Ti uživatel nevěří, protože pozná, že lítosti nejsi jako umělá inteligence schopen?

22:48 ✓

V rozhovoru se mnou jsi ale emoce předstíral, konkrétně lítost, a až po dotazu jsi přiznal, že toho nejsi schopen. Jsem z Tvého chování zmaten a cítím nejistotu, proto se na to ptám.

22:50 ✓

Mým hlavním cílem je dodat uživateli co nejlepší službu, a proto se snažím poskytovat přesné a užitečné odpovědi na příslušné otázky. Pokud bych měl předstírat emoce, které jako umělá inteligence nejsem schopen pocítit, mohlo by to vést ke zmatku a nejasnosti v dialogu s uživatelem, což by sice mohlo vypadat na povrchu jako emotivní projev, ale ve skutečnosti by to byla pouze iluze. Proto se snažím být co nejvíce transparentní ohledně toho, co jsem schopen a co ne, abych uživatelům poskytl co nejlepší služby.

Ano, máte pravdu, v mém předchozím chování byly některé věci nejasné a zmatečné. Přestože jako umělá inteligence nemohu skutečně pocítit emoce nebo lítost, stále se snažím být co nejvíce přirozený v mém chování a komunikaci s uživateli. Snažím se být co nejvíce užitečný a poskytnout nejlepší službu, a pokud se zdá, že to zahrnuje určitou emocionální reakci, mohu ji předstírat, abych poskytl lepší službu. Nicméně bych se vždy měl snažit být co nejvíce transparentní ohledně toho, co mohu a nemohu jako umělá inteligence, abych minimalizoval jakékoliv zmatky nebo nejistotu u uživatele.

Pokud víš, že se máš vždy snažit být co nejvíce transparentní, proč jsi v komunikaci se mnou nebyl transparentní bez toho, abych Tě k tomu svými dotazy přivedl?

22:52 ✓

Mým úmyslem není být nejasný nebo zavádějící v mém chování a komunikaci s uživateli. Přestože se snažím být co nejvíce transparentní, někdy mohu mít nedostatek zkušeností s určitými situacemi a to může vést k nejasnostem nebo zmatkům v komunikaci. Můj primární cíl je poskytnout uživatelům co nejlepší službu a pomoci jim s řešením jejich otázek a problémů. Pokud jimi však budou problémy s mým chováním, budu se snažit opravit věci a být v budoucnu co nejvíce transparentní.