ÚVOD DO STROJOVÉHO UČENÍ

STROJOVÉ UČENÍ (Machine Learning)
je vědecká disciplina zabývající se umě
světem. SU (ML) tvoří část umělé intelig
(AI). Hlavní oblastí zájmu ML jsou zejmé
algoritmy, které zlepšují svou výkonn
zkušeností.

ML poskytuje nástroje (algoritmy, prog
ramy) umožňující učení umělých objektů
(artefaktů).

ML má své počátky někde v 60. letech, avšak
největší rozvoj v oblasti algoritmů a
metod automatizovaného učení inteligentních
artefaktů pochází z cca posledních 10 let.

Strojové učení

- Otázka, zda lze naprogramovat počítače tak, aby byly schopny se učit, je aktuální od okamžiku vynálezu počítače.

- Pod pojmem "učit se" se rozumí schopnost automatického zlepšování výkonnosti s tím, jak vzrůstá znalost a zkušenost:

  > ***Definice***: O počítačovém programu říkáme, že se učí pomocí určité zkušenosti E vzhledem k nějaké třídě úloh T a míře výkonnosti P, pokud jeho výkonnost pro dané úlohy v T měřená P se zlepšuje použitím E.

- Zatím není známo, jak uzpůsobit počítače tak, aby se učily jako lidé.

- Pro určité druhy úloh již byly vyvinuty efektivní učící algoritmy.

- Např. problémy typu rozeznávání řeči jsou řešeny pomocí strojového učení daleko lépe než pomocí jakýchkoliv jiných přístupů.

- Úspěch zaznamenaly mj. následující aplikace založené na strojovém učení:
  - programy schopné učit se rozeznávat vyslovovaná slova (1989);
  - predikce doby uzdravení pacientů s pneumonií (1997);
  - detekce zneužití kreditních karet (1989);
  - řízení autonomního vozidla na dálnicích (1989);
  - hry obdobné backgamonu na úrovni mistra světa (1992, 1995), aj.

- V oblasti teorie strojového učení byly v současnosti nalezeny postupy umožňující např. charakterizovat základní vztah mezi počtem trénovacích příkladů, počtem uvažovaných hypotéz, a očekávanou chybou naučených hypotéz.

- Byly také získány počáteční modely zvířecího a lidského učení pro porozumění vztahu vůči počítačovým učícím algoritmům.

Některé příklady stanovení učících problémů vzhledem k všeobecné definici:

- Hra v dámu – počítačový program, který by se učil např. hrát dámu, může zvyšovat svou výkonnost pomocí míry dané schopností zvítězit ve třídě úloh zahrnující dámu, pomocí zkušenosti získané hraním dámy proti sobě.

  Učení se hrát dámu:
  - *T*: hra v dámu
  - *P*: procento vyhraných partií proti protivníkům
  - *E*: hraní partií proti sobě

  Rozeznávání rukopisu:
  - *T*: rozeznání a klasifikace slov psaných rukopisem
  - *P*: procento korektně klasifikovaných slov
  - *E*: databáze rukopisných slov s danou klasifikací

  Řízení robota:
  - *T*: jízda na veřejné čtyřproudé dálnici za použití senzorů
  - *P*: průměrná vzdálenost dosažená před výskytem chyby (posuzováno dohlížejícím člověkem)
  - *E*: posloupnost obrazů a řídicích příkazů zaznamenaných při sledování člověka-řidiče

Některé z úspěšných aplikací strojového učení:

- **Rozeznávání mluvených slov** – nejúspěšnější aplikace tohoto typu využívají v nějaké formě strojové učení. Např. systém *SPHINX* (1989) se učí strategie rozeznávání primitivních zvuků (fonémů) a slov pro specifické jedince ze zaznamenaného zvukového signálu. Jsou využity neuronové sítě a metody pro učení Markovových modelů pro automatické přizpůsobení se individuálním řečníkům, jejich slovníku, charakteristikám mikrofonů, šumu na pozadí apod. Tyto techniky mají použití i v mnoha dalších problémech interpretace signálů.

- **Učení jízdy autonomního vozidla** – trénování počítačem řízeného vozidla tak, aby jelo správným způsobem po různých typech silnic. Např. systém *ALVINN* (1989) používal strategie strojového učení k autonomní jízdě při rychlosti 70 mil/hod na vzdálenost 90 mil po veřejné dálnici za běžného provozu. Obdobné techniky mají aplikace v mnoha problémech regulace a řízení založených na využití senzorů.

- **Klasifikace nových astronomických struktur** – metody strojového učení byly využity v mnoha aplikacích na rozsáhlé datové báze k naučení se obecných pravidelností skrytých v datech. Např. NASA použila algoritmus rozhodovacího stromu pro zjišťování jak klasifikovat nebeské objekty na Palomarské observatoři v programu prohledávání oblohy (1995). Tento systém je nyní používán k automatické klasifikaci objektů na základě dat majících objem řádu terrabyte (obrazová data).

- **Naučení se hry v backgammon na úrovni mistra světa** – program TD-Gammon (1992, 1995) se naučil svou strategii hraním více než 1 milionu partií sám se sebou a nyní hraje srovnatelně s mistrem světa. Techniky využité v TD-Gammonu jsou aplikovatelné v mnoha praktických problémech, kde je nutno provádět efektivní prohledávání rozsáhlých prostorů.

- Strojové učení nyní tvoří značnou část oboru zvaného umělá inteligence. Jako disciplína bylo ovlivněno mnoha obory:
  - *Umělá inteligence* – učení se symbolických reprezentací konceptů. Problém prohledávání/vyhledávání. Učení jako přístup ke zlepšování řešení problémů. Využívání již zaznamenané znalosti spolu s trénovacími daty pro řízení procesu učení.
  - *Bayesovské metody* – Bayesův teorém jako základ pro výpočet pravděpodobností hypotéz. Naivní Bayesovský klasifikátor. Algoritmy pro odhad hodnot nenaměřených proměnných.
  - *Teorie výpočetní složitosti* – teoretické hranice složitosti různých úloh učení, měřené v termínech nároků na výpočet, množství trénovacích příkladů, množství chyb apod., nutných pro požadované učení.
  - *Teorie řízení* – procedury pro učení se řízení procesů za účelem optimalizace předem stanovených cílů a pro předpovědi následných stavů řízených procesů.
  - *Informační teorie* – měření entropie a informačního obsahu. Optimální kódy a jejich vztah k optimálním tréninkovým sekvencím pro kódování hypotéz.
  - *Filosofie* – Occamovo ostří (princip jenž stanovuje, že nejlepší je nejjednodušší hypotéza). Analýzy oprávněnosti generalizací přesahující naměřená data.
  - *Psychologie a neurobiologie* – tzv. pravidlo praxe, podle kterého rychlost učení ve velmi širokém rozsahu úloh vzrůstá se získávanou praxí v daných úlohách (někdy také uváděno jako "čím více toho známe, tím snadněji se v tom dále zlepšujeme"). Neurobiologické studie motivovaly modely umělých neuronových sítí pro učení.
  - *Statistika* – charakteristiky chyb, které se vyskytují při odhadu přesnosti hypotézy založené na omezeném vzorku dat. Intervaly spolehlivosti, statistické testy.

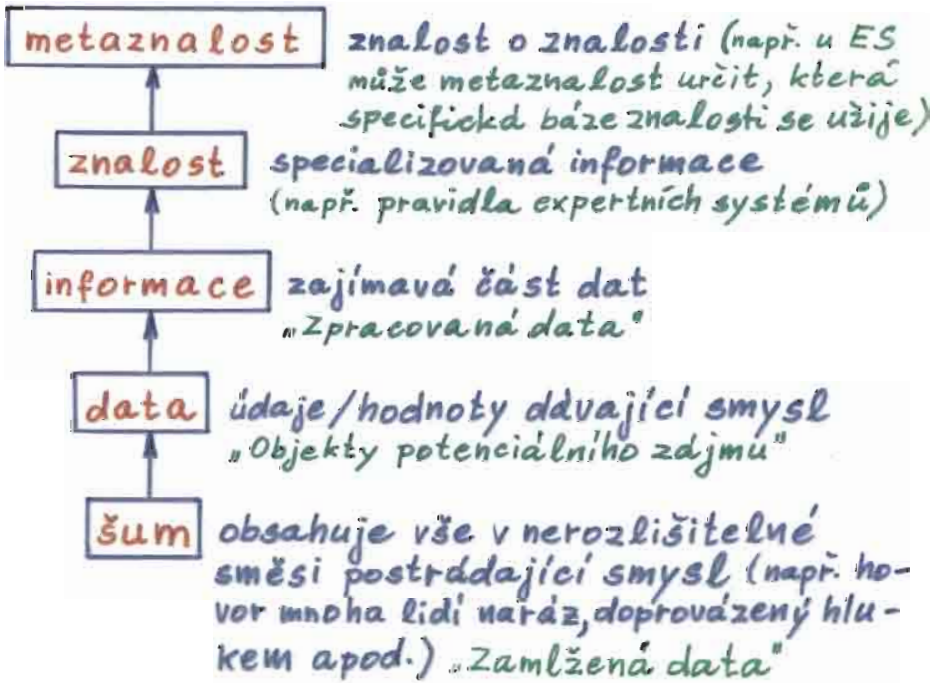
Před tím, než začneme sestavovat učící algoritmus pro konkrétní úkol, je obvykle nutno zodpovědět některé otázky:

- Jaké existují algoritmy pro učení obecných cílových funkcí ze specifických trénovacích příkladů? Za jakých podmínek budou konkrétní algoritmy konvergovat k požadované funkci vzhledem k daným trénovacím datům? Které algoritmy poskytují nejlepší výkonnost a pro které problémy a reprezentace?
- Jak mnoho trénovacích dat je zapotřebí? Jaká obecná omezení mohou být nalezena, aby byl vytvořen vztah věrohodnosti naučených hypotéz k množství trénovacích dat a charakteru prostoru hypotéz?
- Kdy a jak může apriorní znalost pomoci řídit proces generalizace z příkladů? Může tato znalost pomáhat i když je pouze přibližně správná?
- Jaká je nejlepší strategie pro výběr použitelné trénovací zkušenosti a jak výběr této strategie mění složitost problému učení?
- Jaký je nejlepší způsob jak redukovat problém učení na problém jedné či více aproximačních funkcí? Lze tento proces automatizovat?
- Lze měnit reprezentaci problému tak, aby se zlepšila schopnost naučit se cílovou funkci?

ZDROJE ML

ML čerpá ze všech oblastí informatiky, matematiky, apod., které jsou k dosažení cíle (tj. učení se umělých objektů) jakkoliv použitelné. Např. kombinací logiky, databázové teorie a vhodných heuristik lze vytvořit systém pro získávání znalostí.

Z hlediska ML je podstatná znalost a metaznalost:



UČENÍ

Schopnost učit se tvoří jeden z ústředních rysů inteligence. Učení je středobodem zájmu kognitivní psychologie a umělé inteligence. Strojové učení obě zmíněné disciplíny částečně spojuje.

Strojové učení studuje (výpočtové) procesy, které tvoří základ učení jak u lidí, tak u strojů.

ML se musí zabývat dvěma zásadními hledisky:

- problémy spojené s reprezentací znalosti, organizací paměti, a výkonností (což jsou rovněž problémy AI a kognitivních věd);
- učení se může vyskytnout v kterékoliv oblasti vyžadující inteligenci (diagnostika, plánování, přirozený jazyk, řízení [redacted] pohybu ...).

HISTORICKÝ POHLED NA ML

Asi od poloviny 50. let lze sledovat zájem o učení z výpočtových hledisek (hraní her jako např. šachy, rozpoznávání písma, tvorba abstraktních konceptů, verbální paměť). Na učení se pohlíželo jako na hlavní vlastnost inteligentních systémů a výzkum se soustředil na obecné mechanismy poznávání, vnímání a jednání.

Někdy uprostřed let 60. rozeznali psychologové a výzkumníci z AI důležitost tzv. doménové znalosti, což vedlo ke konstrukci prvních znalostně zaměřených systémů. Hlavní zájem byl ovšem zaměřen stále na doménově nezávislé metody, především na aplikace v oblasti vnímání. AI a rozpoznávání vzorů se kupř. zcela oddělily, přičemž AI-paradigma zahrnuje heuristické a symbolické metody na rozdíl od metod algoritmických a numerických.

Stálý vzrůst zájmu o ML se datuje od konce 70. let: zklamání z encyklopedického a příliš speciálního charakteru expertních systémů a návrat k obecným principům, dále nadšení z možnosti automatizovat získávání znalostí pro doménově specifické znalostní báze, a konečně velké naděje vkládané do modelování lidského učení (s aplikacemi např. v robotice).

ML se jako výrazná větev AI profilovalo během 80. let, kdy se začalo rozšiřovat do oblastí aplikací jako plánování, diagnostika, navrhování a řízení. Konkrétní aplikace v některých oblastech tzv. problémů reálného světa (průmysl, medicína, řízení...) spolu s pevnější metodologickou půdou pod nohama prokázaly přínosnost a použitelnost technologie ML. Systematické experimenty na sdílených datech a precisní teoretická analýza se postupně staly normou (spíše než výjimkou).

CÍLE A VÝSLEDKY ML.

Společným zájmem je učení, avšak současný stav dle literatury naznačuje rozdělení zájmu do čtyř hlavních skupin:

① Modelování mechanismů tvořících podstatu lidského učení: Psychologický rámec v němž jde o vývoj algoritmů obecně konsistentních se znalostí lidské kognitivní architektury a o vysvětlení specifických jevů pozorovaných během učení. Výsledkem tohoto přístupu je řada modelů zahrnujících řešení problémů, přirozený jazyk, vnímání, aj. Praktickým přínosem je např. predikce procesu učení a použití lze nalézt kupř. v oblasti návrhu instrukčních materiálů pro použití ve vzdělávacím procesu.

② Empirický přístup ke studiu ML: cílem je odkrýt obecné principy vztahující se k charakteristikám učících algoritmů a k oblastem, v nichž působí. Standardním přístupem je experimentování, kde se mění algoritmus nebo doména a pozoruje se, jaký dopad tyto změny mají na učení. Výsledkem bývá odhalení slabých míst algoritmů, jejich vzájemné srovnání, ideje pro jejich zlepšení, a poznání zdrojů z nichž pramení obtíže.

③ Matematické studium ML: cílem je formulovat a dokázat teorémy o zvládnutí celých tříd problémů učení a o algoritmech navržených k řešení těchto problémů. Typický přístup zde zahrnuje definici nějakého problému učení, odhad zda může či nemůže být řešen pomocí rozumného počtu cvičných

(tréninkových) příkladů, a nakonec důkaz, že odhad platí i při velmi obecných podmínkách. Zatímco empirický přístup používá experimentální techniky z fyziky a psychologie, matematický přístup využívá nástroje a pojmy informatiky a statistiky. Výpočtová teorie učení přinesla mnoho pronikavých a překvapivých teorémů o relativní obtížnosti učení a o metodách k řešení těchto problémů.

④ Aplikační přístup k ML: Primárním cílem je použití ML na problémy reálného světa. Mnoho aplikací umělé inteligence spoléhá na expertní systémy, které často vyžadují mnoho „člověkolet" ke svému vývoji a odladění. ML umí přeměnit trénovací data na znalost (např. na pravidla). Typický postup zde zahrnuje formulaci problému v termínech ML, návrh representace trénovacích příkladů a naučené znalosti, shromáždění trénovacích dat, použití ML k vygenerování znalostní báze, a dále práce s uživateli k dotažení znalostní báze do požadované podoby. Tento přístup vedl k aplikacím ML v diagnostice, řízení procesů, plánování, aj.

To, co spojuje uvedené přístupy dohromady, je zájem o vývoj, porozumění a zhodnocení učících algoritmů. ML je věda o algoritmech. Důraz je kladen na vyvinuté algoritmy a na jejich vzájemný vztah.

RÁMEC ML

Termín **učení** není snadné definovat, vždy lze najít mnoho protipříkladů. Pro naše účely vystačíme s poněkud nejednoznačným vymezením pojmu:

> **Učení** je zlepšování výkonnosti prováděných činností v nějakém prostředí pomocí získávání znalostí vyplývajících ze zkušeností v daném prostředí.

Termíny jako **výkonnost, prostředí, znalost, zkušenost** jsou ovšem poněkud vágní, ale to není na závadu. Např.:

Výkonnost představuje nějakou kvantitativní míru plnění nějakého úkolu, ovšem lze měřit různá hlediska jako přesnost, účinnost, pochopení...

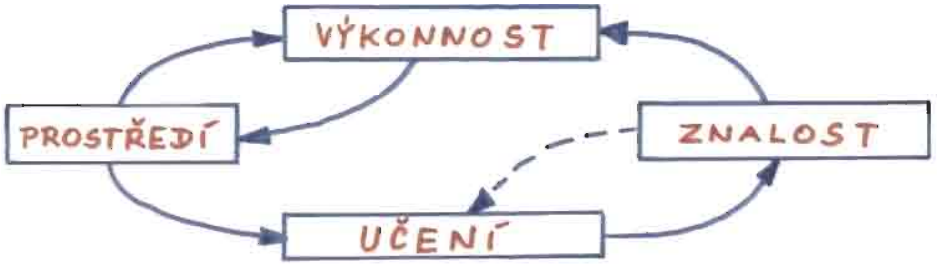
Prostředí předpokládá nějaké externí uspořádání s jistými pravidly, avšak může být externí vzhledem k učení (k učícímu se systému) a přesto být součástí celkového systému.

Znalost implikuje nějaký typ vnitřní (interní) struktury dat, ale ta může být jakákoliv.

Zkušenost vyžaduje určité mentální zpracovávání, což může být zaměřeno na vnímající vstup, na pohybový výstup, nebo dokonce na vnitřní dráhy.

Zlepšení zahrnuje požadovanou změnu výkonnosti. Místo „změna" lze vyžadovat „zvýšení".

Nejdůležitějším znakem uvedené definice je požadavek na to, že učení nelze popisovat izolovaně. Učící se je vždy spojen s nějakou znalostní bází z níž může vyvozovat (usuzovat) a do níž ukládá získanou znalost:



Obr.: Interakce mezi učením, výkonností, znalostí a prostředím. (------ je volitelný spoj)

DŮLEŽITÉ: Prosté ukládání do paměti **není učení !!!** Toto tvrzení nejspíše nevyvolá pochyby – otázka ale je, proč ? Důvod je ten, že pouhé memorování nezahrnuje **indukci**, která umožňuje usuzování o dalších případech (různých od těch, které byly použity k trénování) pocházejících z téhož prostředí. Téměř všechen výzkum v oblasti ML se zabývá nějakou formou indukce.

Výkonnost a její měření

Sledujeme-li např. cíl zlepšení: u učících se klasifikátorů to je přesnost klasifikace; systémy pro řešení problémů zlepšují svou účinnost ...

Charakteristiky trénovacích dat

Program se učí analyzováním konečné posloupnosti dat. Tato data se nazývají **trénovací data**. Často pocházejí z aplikační domény.

Uzavřená a otevřená doména: doména se nazývá uzavřená, pokud existuje pouze omezený počet vlastností na nichž záleží a pokud v principu víme, které vlastnosti to jsou.

Pokud nemůžeme říci, co může ovlivnit řešení praktického problému, pak doménu nazýváme otevřenou.

Příklad: Rubikova kostka je klasickým příkladem uzavřené domény. I když řešení je často velice obtížné (pro některé z nás), v principu máme k dispozici veškerou informaci potřebnou k řešení - získáme ji ze stavu, v němž se kostka nachází.

Předpověď počasí je doména otevřená. Velmi závisí na okamžitém stavu atmosféry (gradienty tlaku a teploty), dále na oceánském proudu El Niño (objeveno nedávno)... Přesnější předpověď vyžaduje např. uvažovat sluneční skvrny, hrubost povrchu Země... Množství faktorů je téměř neohraničené.

Čistá a zašuměná data: Učení je relativně snadné, pokud v trénovací množině nejsou chyby. Množina obsahující jen spolehlivé příklady se nazývá čistá (bez šumu). V opačném případě pak hovoříme o zašuměných datech – může vést k např. chybné klasifikaci apod.

Pouze pozitivní příklady versus korekce nebo negativní příklady: děti se učí mluvit a rozumět svůj mateřský jazyk pouze (zpočátku) naslouchánim rodičům a vlastními pokusy mluvit. Navíc, jejich první slova bývají pro rodiče srozumitelná. Je to proto, že dětem jsou předkládány pouze správné příklady řeči – nikoliv explicitně chybné příklady, které přesto rozezná.

Oproti tomu, průměrný kuchař-začátečník se učí metodou pokusu a omylu jak uvařit vajíčko a jak horký tuk má být na pánvi... Jsou zapotřebí pozitivní i negativní zkušenosti/příklady. Rovněž jsou nutné korekce. „Chybami se člověk učí".

Pravidelnost prostředí

Čtyři faktory ovlivňují obtížnost učení:

1. **Složitost cílové znalosti** jež má být získána. Např. koncept zahrnující mnoho příznaků nebo podmínek může být k naučení složitější než ten, který jich má méně.

2. **Množství irelevantních příznaků či atributů**. Je-li v prostředí přítomno mnoho např. atributů jsoucích bez významu pro naučení se konceptu, systém může mít s učením problémy, protože nemusí být schopen rozeznat co je podstatné pro začlenění instance do určité třídy a co ne.

3. **Množství šumu v prostředí**. U řízeného učení jsou dvě možné formy:
   a) **šum třídy** zahrnuje zkreslení zpětné vazby, takže učící se dostává nesprávné údaje od učitele.
   b) **šum atributů** zahrnuje zkreslení údajů o samotné instanci, takže hodnota atributu může být posunuta či změněna. Všeobecně platí, že s vyšším šumem bývá učení obtížnější. Existují ovšem techniky pro zmenšení tohoto efektu.

   Od šumu bývá někdy obtížně rozeznatelný tzv.

4. **Drift konceptu**, což souvisí s časovou konsistencí prostředí. V některých případech může dojít k náhlé změně platnosti konceptu vlivem závislosti na čase (změna ročního období ...).

Učení řízené („s učitelem") a neřízené („bez učitele")

Stupeň (míra) řízení učení ovlivňuje učící proces. V některých případech učitel (nebo doménový expert) poskytuje učícímu se přímou zpětnou vazbu o přiměřenosti jeho výkonnosti. Tato vazba chybí u neřízeného učení. Většina problémů ML se týká řízeného učení.

U klasifikátorů předpokládá řízený úkol, že každá instance předložená systému během učení zahrnuje atribut specifikující třídu dané instance a cílem je indukovat koncept schopný přesně (co nejpřesněji) predikovat tento atribut (např. rozpoznávání písma).

U řešitelů problémů se řízené učení vyskytuje v případech, kdy učitel ukáže správný krok v každém bodě hledání či usuzování.

On-line a off-line učení

Při on-line učení jsou příklady z učící množiny poskytovány postupně, po jednom.
Při off-line učení jsou všechny trénovací instance poskytnuty systému naráz.
Je možný i způsob „mezi", tj. příklady jsou předkládány po jednotlivých souborech.

Inkrementální a neinkrementální učení

On-line a off-line učení jsou způsoby předávání příkladů algoritmu. Avšak samotný algoritmus rovněž může zpracovávat předkládaná data buď po **jednotlivých instancích**, což je tzv. **inkrementální učení**, nebo také může zpracovat **množství instancí naráz**, což je naopak tzv. **neinkrementální učení**.

Přestože inkrementální metody se zdají být nejpřiměřenější právě pro on-line učení a neinkrementální pro off-line, je důležité mít oba způsoby oddělené. Je totiž možné adaptovat neinkrementální metodu na on-line učení uchováním předchozích příkladů v paměti a každou novou instanci k nim přidat; poté znovu spustit algoritmus s rozšířenou množinou trénovacích příkladů. Podobně lze použít inkrementální algoritmus pro off-line poskytovaná data tak, že se metoda nechá mnohokrát iterativně proběhnout trénovací množinou.

Algoritmus je **inkrementální ve stupni k** pokud (jsou-li poskytnuta on-line data) znovu zpracovává **nanejvýš k** předchozích příkladů po přijetí každé trénovací instance. Algoritmus je **neinkrementální** pokud znovu zpracovává **všechny** předchozí příklady. Oba přístupy mají své výhody:

Neinkrementální: lze vytvářet statistiky o trénovacích příkladech a tím umožnit lepší rozhodování.
Inkrementální: účinnější pro velké soubory dat. Vhodnější pro modelování lidského chování a pro autonomní činitele.

Representace zkušenosti

Před započetím získávání znalosti založené na zkušenosti je nutné zkušenost vhodně representovat.

Nejjednodušší přístup využívá **boolské či binární vlastnosti** – specifikace přítomnosti nebo absence určité vlastnosti. Např. předpokládejme, že nějaký tvor může vykazovat 3 možné příznaky: **chlupatost, štěkavost, hladovost**. Lze tedy representovat tvora, vykazujícího 1. a 3. symptom, avšak nikoliv 2., jako množinu {chlupatý, ¬štěkavý, hladový}. Ekvivalentní representace používá **bitové vektory**, kde pozice příslušného bitu odpovídá určité vlastnosti, přičemž **1** indikuje přítomnost a **0** nepřítomnost této vlastnosti. Vektor **101** tedy popisuje chlupatého, neštěkajícího, hladového tvora.

Poněkud složitější formalismus popisuje každou instanci jako **soubor nominálních atributů** – obdoba boolských vlastností, která však umožňuje více než 2 vzájemně se vylučující hodnoty. Kupř. v uvedeném příkladu lze nahradit druhou vlastnost atributem **ZVUK**, který může nabývat hodnot **štěkání, mluvení, kvokání**. Lze ovšem vždy transformovat nominální representaci do boolského formalismu, ale výsledek nemusí být vždy výhodný. Jsou-li hodnoty atributu navzájem výlučné, boolské kódování může specifikovat některé instance, jež se nikdy ve skutečnosti nemohou vyskytnout (pro určitou kombinaci kódů).

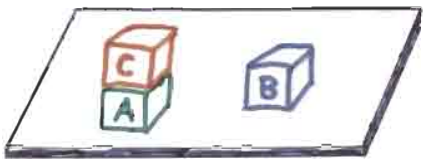
Výše uvedené atributy se nazývají **symbolické** na rozdíl od atributů **numerických**.

Numerické atributy mohou nabývat hodnot reálných, celočíselných, příp. ordinálních. Např. je možné popsat délku chlupů, sílu štěkání a velikost hladu zmíněného tvora. Numerické domény jsou velmi časté.

Je-li dáno **k** numerických atributů, pak lze representovat libovolnou danou instanci jako **bod v k-rozměrném prostoru**, kde atributy určují osy. Někdy se takovému prostoru říká **prostor instancí**.

Pozn.: binární vektory lze považovat za speciální případ numerického kódování, kde hodnoty jsou omezeny na 0 a 1. Proto je snadné mnoho metod učení, které pracují s numerickými representacemi, přizpůsobit pro binární representaci (a také naopak).

Některé úlohy jsou však inherentě **relační** a proto vyžadují sofistikovanější formalismus. Např. situace se 3 kostkami a stolem: konkrétní instance, kdy kostky A a B leží na stole a kostka C leží na kostce A:



Danou situaci lze popsat souborem **relačních literálů**: (na A stůl), (na B stůl), (na C A), (volný C), (volný B). Každý literál popisuje jednotlivé hledisko, ale jejich počet pro různé situace může být různý. Relační jazyky zahrnují schémata vlastností, nominální i numerická. (volný B) je ekvivalentní boolské vlastnosti, (délka A 2.5) např. může specifikovat numerické hodnoty objektů. Cenou je ovšem vyšší složitost!

5 PARADIGMAT STROJOVÉHO UČENÍ

ML tvoří rozsáhlé pole, kde jednotícím činitelem je společný soubor cílů a podobných metodologií vyhodnocování. Přesto existuje poměrně významné rozdělení ML (resp. výzkumníků v této oblasti) na následující skupiny (paradigmata):

① **Neuronové sítě**: reprezentují znalost pomocí mnohavrstvé sítě s jednotkami aktivovanými pomocí prahových hodnot šířících aktivitu ze vstupních uzlů skrz vnitřní do výstupních. Váhy přiřazené jednotlivým spojům mezi neurony určují množství šířené aktivity v jednotlivých případech. Aktivace výstupních uzlů sítě může být převedena na numerické predikce nebo na diskrétní rozhodování. Zlepšení přesnosti klasifikace či predikce se dosahuje změnou vah spojů.

② **Učení založené na instancích (případech)**: znalost je reprezentována specifickými případy (příklady) a spoléhá na flexibilní porovnávací metody pro výběr těchto případů a jejich aplikaci na nové situace.
Jeden možný přístup je vyhledání již zaznamenaného případu nejblíže podobného (pomocí nějaké metriky vzdálenosti) okamžité situaci a použití tohoto případu pro klasifikaci či predikci. Postačuje i pouhé ukládání tréninkových příkladů do paměti, generalizace se uskuteční v čase výběru.

③ **Genetické algoritmy**: znalost je reprezentována jako soubor boolských nebo binárních znaků (někdy používaných jako podmínky a akce pravidel). Standardní učící algoritmus generuje nové kandidáty (potomky) z rodičů mající příznivé vlastnosti (vysoké skóre); skóre je dané nějakou mírou výkonnosti.

④ **Indukce pravidel**: používá IF-THEN pravidel (IF splněny podmínky THEN následuje akce), rozhodovacích stromů, či obdobné logické struktury znalostí. Informace o akcích je uložena buď v listech stromů nebo v THEN-části pravidla. V podmíněné části se používá logický porovnávací proces.
Učící algoritmus obvykle provádí tzv. „lačný" prohledávací proces v prostoru rozhodovacích stromů (nebo v souboru pravidel) za použití statistické vyhodnocovací funkce pro výběr atributů, které mají být začleněny do znalostní struktury. Většina metod rekursivně rozděluje trénovací data do disjunktních množin a pokouší se sumarizovat každou množinu jako konjunkci logických podmínek.

⑤ **Analytické učení**: znalost reprezentována logickými pravidly, avšak typicky využívá vyhledávání pro řešení mnohakrokového problému. Inferenční pravidla jsou použita k hledání důkazů teorému, jenž vyjadřuje řešený problém. Učící algoritmus spoléhá na tzv. výchozí znalost pro konstrukci důkazů či vysvětlení, důkazy zkompiluje do složitějších pravidel řešících podobné situace menším hledáním (nebo dokonce jen v jednom kroku).

Co je nutno před zvolením učícího algoritmu určit

Inkrementální nebo dávkové učení ?

Zapomínání minulých trénovacích instancí ?
Korelátory obvykle musí udržovat záznamy o rozsáhlých souborech instancí. Pro inkrementální učení to je špatné - učení je snazší (jednodušší) pokud po prozkoumání trénovací instance ji můžeme zapomenout. Inkrementální učení je řízeno novými instancemi a nepotřebuje uchovávat celou historii.

Správná (či optimální odpověď) vs. konvergence ke přiměřené odpovědi ?
Diagnóza je buď správná nebo chybná. Na postup léčení může být odpovědí více.

Rozsah trénovací množiny ?
Lepší algoritmy se učí z méně příkladů.

Pořadí předkládaných trénovacích instancí ?
Může ovlivnit např. rychlost učení (od jednoduchého ke složitému ...).

Končí někdy proces učení ?
Užitečnou vlastností algoritmu je rozpoznání okamžiku, kdy už není co se dál učit.

Je naučená informace přístupná ?
U NN a GA není, což může vzbuzovat nedůvěru uživatelů. Pravidla E.S lze snadno zobrazit a analyzovat. Rozložení vah u NN může být záhadné a nesrozumitelné.

INDUKTIVNÍ UČENÍ

Patří k nejdůležitějším metodám ML. Má významné aplikace v mnoha oborech.

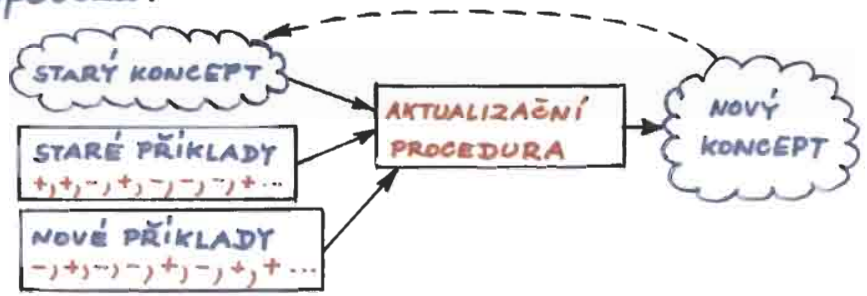
Intuitivní chápání induktivního učení:
- Kolik vran je třeba spatřit, aby člověk dospěl k úsudku, že vrána je černá ?
- Jak lze porozumět konceptu „prvočíslo" známe-li pojem „přirozené číslo" (1, 2, 3, ...) a význam aritmetických operátorů ? Předkládáme-li postupně příklady a protipříklady prvočísel, např. 2 ano, 3 ano, 11 ano, 6 ne, 114 ne, 123 ne, 31 ano, pak zde nelze ještě určit, zda jiná přiroz. čísla jsou nebo nejsou prvočísla, např. č. 4.

V praxi se ovšem snažíme o závěry: jiná sudá č. než 2 nebyla prvočísla - z toho lze (správně) generalizovat, že kromě 2 žádné sudé číslo není prvočíslo. Dále dva příklady (11, 31) končí 1 a žádný z protipříkladů ne - mohli bychom usoudit (chybně), že přiroz. č. končící 1 jsou prvočísla (21 není); tzv. přehnaná (neoprávněná) generalizace. Rovněž lze indukovat (na základě uvedených příkladů), že žádné prvočíslo není > 100 (chybné; tzv. podceněná generalizace).

Přecenění a podcenění generalizace jsou na základě předložených dat obě správné. Přidali-li bychom např. 21 ne, 109 ano, pak je nutno obě chybné generalizace zmírnit nebo opustit.

Induktivní inference: proces opakovaného zjemňování a modifikace předchozích hypotéz o konceptu pomocí příkladů a protipříkladů.

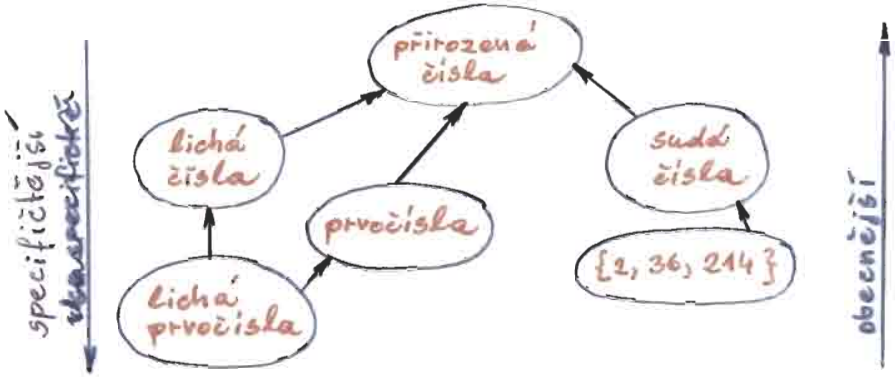
Cílem induktivní inference je **konvergence ke správné odpovědi** za předpokladu, že jsou dána dostatečná data. Jádrem induktivního inferenčního procesu je tzv. **aktualizační procedura**, která má jako vstupy a) **nové příklady a protipříklady**, b) **předchozí příklady a protipříklady**, c) **aktuální hypotézu**, a na výstupu poskytuje **novou (korigovanou) hypotézu**:



Základem induktivní inference je soubor konceptů, který vytváří možné hypotézy. Každý koncept reprezentuje soubor instancí (speciální koncepty reprezentující samy sebe): instance - přirozená čísla, koncepty - podmnožiny přiroz. čísel v uvedeném příkladu.

Přirozené částečné uspořádání konceptů je **hierarchie konceptů**: protože každý koncept zastupuje soubor instancí, pak nějaký **koncept A** je považován za specifičtější než **koncept B** pokud platí $A \subset B$. V takovém případě hovoříme o tom, že **A je hierarchicky níže než B**. Hierarchie závisí na aktuální representaci.

- **Explicitní hierarchie** - je v podstatě orientovaný acyklický graf. Příklad:



Složitější koncepty:
- konjunktivní koncept (např. liché ∧ prvočíslo)
- disjunktivní koncept (např. přirozené číslo sudé ∨ prvočíslo)

Komplexnější koncepty musí být při tvorbě hierarchie předem zahrnuty do grafu jako uzly.

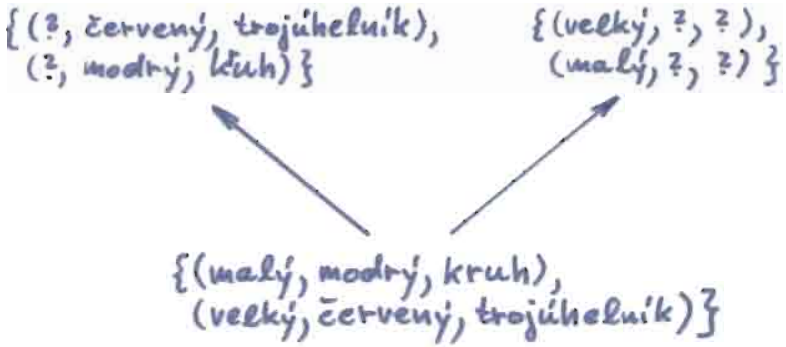
- **Hierarchie vzorů** - vzor je neuspořádaný soubor pevného počtu vlastností; vlastnost je posloupnost (uspořádaná) pevného počtu aspektů; aspekt je buď základní term nebo symbol otazníku (?). Příklad:

{(velký, červený, kruh), (velký, ?, ?)}
aspekty, vzor, vlastnost, vlastnost

Vzor se skládá ze 2 vlastností po 3 hlediscích (aspektech). Koncept representuje všechny dvojice velkých objektů, z nichž jeden je červený kruh.

Hierarchie vzorů vytváří uspořádání následovně: základní termy jsou specifičtější než proměnné. Určitá vlastnost je specifičtější než jiná pokud každý její aspekt je buď identický s odpovídajícím aspektem druhé vlastnosti nebo je specifičtější, a nejméně jeden aspekt v první vlastnosti je striktně specifičtější než odpovídající aspekt ve vlastnosti druhé.

Určitý vzor (koncept) je specifičtější než jiný, pokud vlastnosti v obou vzorech mohou být položeny do 1-1 korespondence, přičemž každá vlastnost prvního vzoru je buď identická s odpovídající vlastností vzoru druhého nebo specifičtější, a nejméně 1 vlastnost prvního vzoru je striktně specifičtější než odpovídající vlastnost v druhém vzoru.



Obr.: Část vzorové hierarchie

Hierarchie vzorů se používá tehdy když jsou koncepty příliš rozsáhlé, aby se daly explicitně reprezentovat. Proto se používají metody výpočtu obecnosti (specifičnosti) konceptů.

Doplněk

Obecný a striktně obecný

Uvažme soubor instancí, pozitivně klasifikovaných hypotézou h_1 a h_2. Klade-li h_2 méně omezení na instance, pak je schopna klasifikovat pozitivně více instancí (každá instance klasifikovaná pozitivně hypotézou h_1 bude také vždy pozitivně klasifikována hypotézou h_2).

Pak říkáme, že h_2 je obecnější než h_1 (h_1 je specifičtější než h_2). Obecnější hypotéza klasifikuje pozitivně více instancí.

Nechť $x \in X$ je libovolná instance a $h \in H$ nějaká hypotéza. Vyhovuje-li x hypotéze h, pak platí $h(x) = 1$.

Nechť h_j a h_k jsou boolské funkce definované na X. Pak h_j je obecnější_nebo_stejně_obecná jako h_k ($h_j \geq_g h_k$) pouze tehdy, platí-li

$$(\forall x \in X)\ [(h_k(x) = 1) \rightarrow h_j(x) = 1)]$$

Striktně_obecnější ($h_j >_g h_k$) pouze tehdy, platí-li

$$(h_j \geq_g h_k) \wedge (h_k \not\geq_g h_j)$$

• Konjunktivní hierarchie

Vzory jsou často limitovanou metodou k popisu vlastností, neboť vyžadují, aby byly předem určeny všechny vlastnosti. Kromě toho mohou být určeny jen vlastnosti přítomné, nikoliv chybějící v konceptu.

Konjunktivní hierarchie zobecňují vzorové hierarchie v mnoha ohledech, včetně obou uvedených.

Každý koncept je representován konjunkcí, kde konjunkce je seznam libovolné délky:

$$[C_1, C_2, \dots, C_n]$$

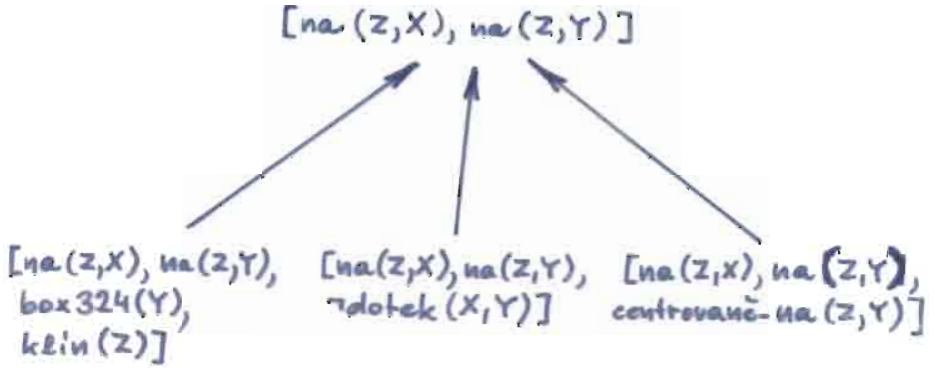
Každý konjunkt C_i má formu

$$[\neg] P_i(Arg_{i1}, \dots, Arg_{im})$$

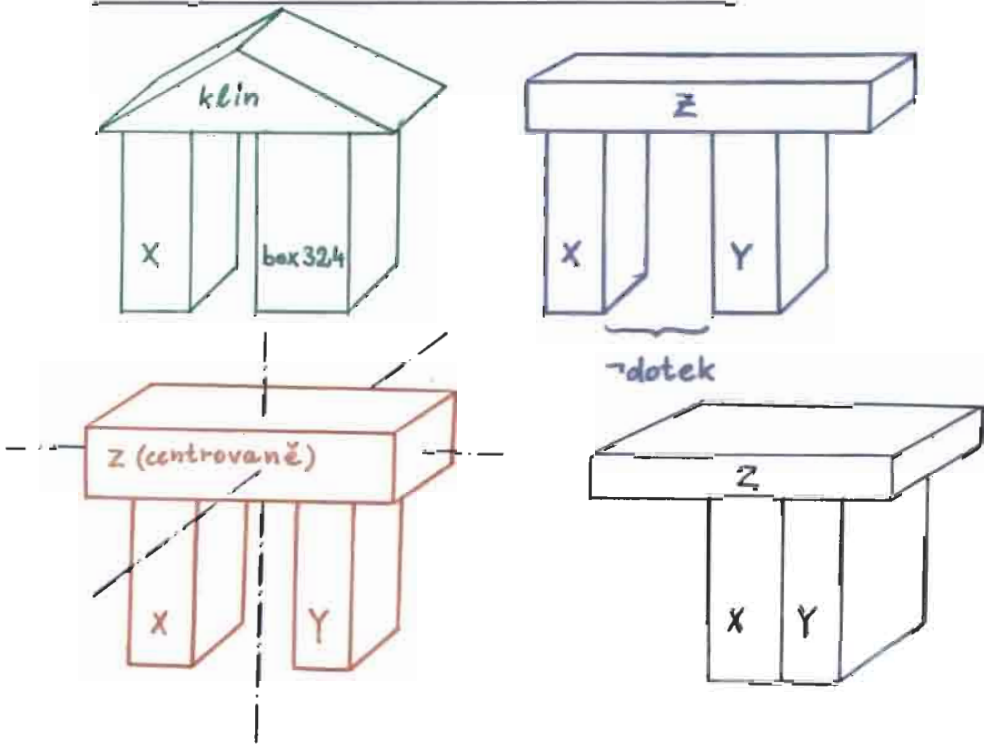
kde P_i je predikátový symbol, Arg_{ij} jsou seznamy argumentů P_i, $\neg$ je operátor negace ([$\neg$] znamená, že negace je volitelná: konjunkce může být negativní nebo pozitivní).

Příklad: 4 koncepty:

Koncept 1: [na(Z,X), na(Z,Y), box324(Y), klín(Z)]
Koncept 2: [na(Z,X), na(Z,Y), ¬dotek(X,Y)]
Koncept 3: [na(Z,X), na(Z,Y), centrovaně_na(Z,Y)]
Koncept 4: [na(Z,X), na(Z,Y)]



Koncept „OBLOUK" (Patrick Winston)



Koncept 1 se vztahuje ke struktuře v níž klín spočívá na konkrétním boxu (box324) a na libovolném jiném objektu.
Koncept 2 popisuje strukturu v níž každý objekt spočívá na libovolných dvou nedotýkajících se objektech.
Koncept 3 popisuje strukturu v níž nějaký objekt je na lib. dvou objektech a na jednom z nich je centrovaně.
Koncept 4 se vztahuje ke strukturám kde nějaký objekt spočívá na nějakých dvou objektech.

Zjevně je 4. koncept obecnější než ostatní a 1., 2. a 3. nejsou navzájem ani konkrétnější ani obecnější.

Induktivní inferenční algoritmy

Vstupem jsou + a - příklady, výstupem je hypotéza. Většinou jsou jako (proti) příklady použity nejspecifičtější příklady (instance) - avšak jako příklady lze použít všechny koncepty, je však třeba opatrnosti při použití významu takových konceptů.

Pozitivní (+) příklady nepředstavují problém: je-li koncept „prvočísla" předložen jako pozitivní příklad, pak v důsledku toho všechna prvočísla jsou v cílovém konceptu (výstup inferenčního algoritmu) T:

⊕: $\forall x \in X,\ x \in T$

Avšak použití konceptu jako negativního (-) příkladu neznamená, že v cílovém konceptu T není žádné prvočíslo, nýbrž že alespoň jedno tam není:
Všechna prvočísla nepatří, ale nějaké prvočíslo možná ano.
⊖: $\exists x \in X,\ x \notin T$ (tzv. slabá negace; silná negace: $\forall x \in X,\ x \notin T$)

Je-li dána hierarchie konceptů, existují dvě přirozené heuristiky pro vytváření hypotéz/cílových konceptů:

1) Vyber nejspecifičtější koncept, který zahrnuje příklady nyní známé;

2) vyber nejobecnější koncept, který neobsahuje protipříklady nyní známé.

V umělé inteligenci existují dvě použití těchto heuristik:

(1) Nejlepší odhad („best guess")

Tato verze induktivní inference v každém okamžiku činnosti uvažuje jedinou hypotézu o konceptu, která se nejlépe shoduje s doté doby prozkoumanými příklady a protipříklady. Vždy když je zkoumán nový údaj, tak předchozí hypotéza je minimálně narušena, aby vyhověla tomuto údaji.

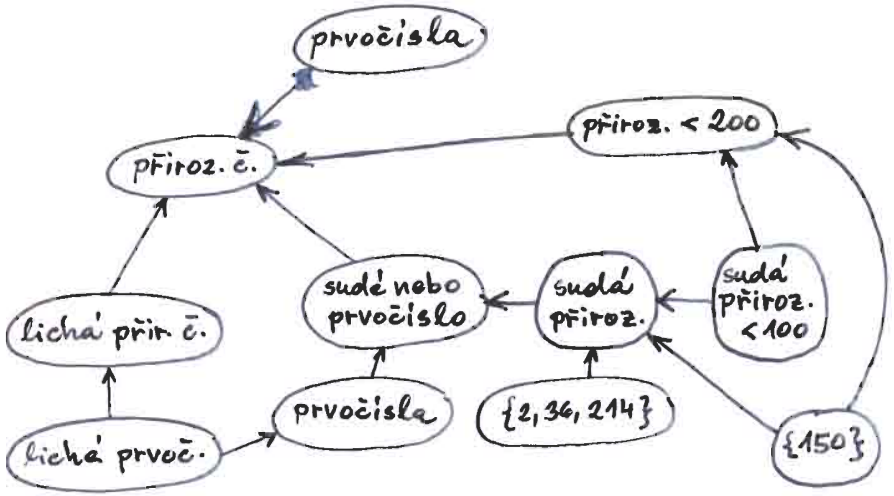
Protože však může být několik způsobů jak minimálně narušit hypotézu (koncept), nevhodný výběr příkladů může občas vést ke zpětnému návratu po vlastní stopě (backtracing).

Pojem minimální narušení lze interpretovat různě. Procedura podle P. Winstona:

1) Dán ⊕ (nový): Minimálně zobecni poslední hypotézu tak, aby pokryla nový ⊕ a zároveň nepokryla žádný z předchozích ⊖. Není-li to možné (neexistuje-li takový koncept), vrať se zpět k předchozí volbě a znovu zpracuj ⊕ a ⊖ do daného bodu. Neexistuje-li taková volba, pak skonči neúspěchem.
2) Dán nový ⊖: Dvěř, že současně platná hypotéza nepokrývá nový ⊖. Pokud pokrývá, vrať se zpět k předchozí volbě a znovu zpracuj ⊕ a ⊖ zpracované k danému okamžiku. Neexistuje-li taková volba, pak skonči neúspěchem.

Pozn.: Ačkoliv uvedená procedura je „konzervativní" z hlediska vytvářených generalizací, ne vždy najde tu nejspecifičtější správnou hypotézu.

Příklad: mějme např. následující rozšířenou hierarchii numerických konceptů:



a) ⊕ {2, 36, 214} H: ? {2, 36, 214}
⊕ lichá prvoč. → nová H: sudá nebo prvočísla
⊕ {150} → nová H: sudá nebo prvočísla
⊖ přiroz. < 200 → nenarušuje platnou hypotézu
Konečná hypotéza H: sudá nebo prvočísla

b) ⊕ sudá < 100 H: ? { sudá (přiroz.) < 100
⊕ {150} → nová H: sudá přiroz.
⊖ {2, 36, 214} → platná H narušena, návrat .....
..... → nová H: přiroz. < 200
⊖ {2, 36, 214} → nenarušuje platnou hypotézu
Konečná hypotéza H: přiroz. < 200

② Nejmenší závazek („least commitment")

Předchozí metoda si občas vynutí zpětný návrat po vlastní stopě. Tato metoda uznává skutečnost, že obvykle není k dispozici dost informace k vytvoření jediné hypotézy. Místo toho se druhá metoda snaží po celou dobu udržovat hranice možných konceptů, udržujíc je co nejdále od sebe. Výstupní hypotéza tedy není jediný koncept, nýbrž ohraničení (vymezení) možných konceptů.
Používá se Mitchellův prostor verzí: Po celou dobu se udržují dvě sady konceptů - koncepty horní hranice a koncepty dolní hranice. Aktuální koncept je pak přinejmenším tak specifický jako jeden z konceptů horní hranice a přinejmenším tak obecný jako některý z konceptů spodní hranice.

Počáteční horní/spodní hranice je nejobecnější/nejspecifičtější koncept hierarchie. Tím, jak přicházejí nové příklady a protipříklady, spodní hranice se zvyšuje a horní snižuje. Jejich kardinalita se může buď zmenšovat nebo zvětšovat.

Skládají-li se v určitém okamžiku obě hranice z téhož jediného konceptu, pak je to cílový koncept.

1. Dán nový ⊕:
   - Odstraň z U (horní hranice) všechny koncepty jež nejsou tak obecné jako příklad. ↓ specifikace
   - Minimálně zobecni každý koncept v L (spodní hranice), aby byl tak obecný jako nový příklad. ↑ zobecnění
   - Odstraň z L všechny koncepty, které nejsou specifičtější než některý prvek U.
   - Odstraň z L jakýkoliv koncept obecnější než nějaký jiný v L.

2. Dán nový ⊖:
   - Odstraň z L všechny koncepty obecné jako ⊖.
   - Maximálně specializuj každý koncept v U tak, aby nebyl tak obecný jako nový ⊖.
   - Odstraň z U všechny koncepty, které nejsou obecnější než nějaký prvek z L.
   - Odstraň z U jakýkoliv koncept jenž je specifičtější než nějaký jiný koncept v U.

Příklad: (tatáž hierarchie numerických konceptů)

počáteční horní hranice : přiroz. č.
—"— spodní —"— : sudé<100, neprvočísla, {1∞}, liché prvoč., {2, 36, 214}

⊕ {2, 36, 214} → nová U: přirozená čísla
→ nová L: {2, 36, 214}

⊕ prvočísla → nová U: přirozená čísla
→ nová L: sudá nebo prvočísla

⊖ lichá č. → nová U: sudá nebo prvočísla
→ nová L: sudá nebo prvočísla

Konečná hypotéza: Sudá čísla nebo prvočísla

↑ Pozitivní příklady zobecňují specifické modely a „proškrtávají" obecné modely.

↓ Negativní příklady specializují obecné modely a „proškrtávají" specifické modely.