


FIT SZZ – NI-ZI 2024

Přepis materiálů níže s upravenými a přidanými vysvětlivkami, něco je naopak stručnější - třeba to někomu pomůže.

Zdroje

-  NI-ZI-SZZ
- <https://github.com/kjanovska/NI-SZZ>
- Přednášky

1 – Automatické plánování, plánovací graf, kompilace plánování do jiných formalismů jako je SAT nebo CSP, hierarchické plánování, plánování v prostoru plánů. Plánování pohybu a problém lokalizace v robotice.

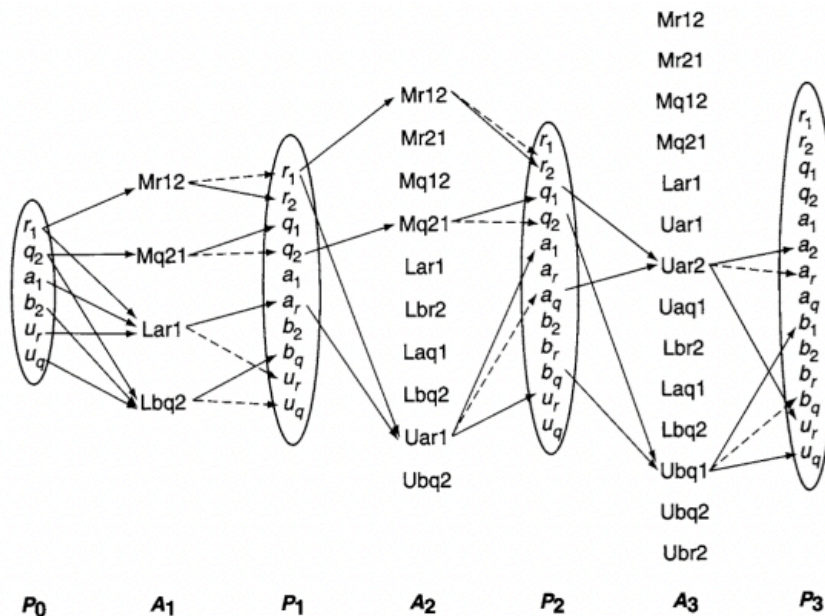
Automatické plánování

- Plánování: hledání posloupnosti akcí vedoucích k dosažení cíle
 - Využití: robotika, doprava, výroba, hry, vesmír
 - Týká se akcí agenta, který koná akce
 - Může být prováděno opakovaně, např. pokud se změní cíl, prostředí, informace od jiných agentů či dojde k chybě
- Typy
 - Klasické
 - Stochastické (pravděpodobnostní): akce je provedena s nějakou pst.
 - Plánování pro dynamické prostředí: nejen agent mění prostředí
- Prostor v klasickém plánování
 - Plně pozorovatelné
 - Deterministické: akce se vždy provede a výsledek je pevně daný
 - Konečné
 - Statické: mění se jen konáním agenta
 - Diskrétní
 - Offline: nejprve plánujeme, pak konáme
- Vyjadřovací prostředky
 - Jazyk: konstanty a predikátové symboly
 - Stav: konečná množina základních atomů (predikátových symbolů s dosazenými konstantami)
 - Svět je uzavřený - co není zmíněno, neplatí
 - Cíl: specifikace stavu (podmnožiny literálů, které musí stav obsahovat)
- Operátor: trojice (ve formátu STRIPS - Stanford Research Institute Problem Solver)
 - name(o): jméno operátoru se seznamem proměnných
 - precondition(o): množina literálů, které musí být splněny pro aplikaci operátoru
 - Mohou být pozitivní i negativní - požadované i zakázané
 - effect(o): množina literálů, které budou platit po provedení
 - Pozitivní přidáváme, negativní odebíráme
- Akce A
 - Instance operátoru, která vzniká substitucí - dosazení konstant za proměnné
 - Aplikovatelná, pokud jsou pozitivní předpoklady podmnožinou stavu a negativní mají se stavem prázdný průnik
- Plánovací problém
 - Plánovací doména: Jazyk L , operátory O
 - Tím implicitně určuje množinu stavů a funkci následníka

- $\gamma: S \times A \rightarrow S$, kde S jsou všechny podmnožiny množ. základních atomů
- Zadání problému: trojice $P = (O, s_0, g)$ - operátory, počáteční stav, cíl
- Plánovací problém: trojice (Σ, s_0, S_g)
 - $\Sigma = (S, A, \gamma)$: stavový prostor s přechody (funkcí následníka)
 - s_0 : počáteční stav
 - S_g : množina cílových stavů (pozitivních a negativních)
- Plán: posloupnost akcí $\pi = [a_1, \dots, a_n]$, kde a_i je instance operátoru z O
 - Je řešením zadání P , pokud je proveditelný z počátečního stavu a dosahuje cíle

Plánovací graf

- Vrstevnatý orientovaný graf
 - Střídání stavových (P_i) a akčních (A_i) vrstev
 - Stavová vrstva obsahuje atomy
 - Akční vrstva obsahuje akce aplikovatelné na předchozí stav
 - Akce pouze přidávají atomy, negativní efekty ignorujeme
 - Hrany propojují předpoklady, akce a jejich pozitivní efekty
 - Negativní akce přerušovanou čarou



- Plánovací graf určuje paralelní plány
 - Sjednocují obrovské množství sekvenčních plánů
 - Lze je převést na sekvenční, pokud jsou akce nezávislé

- Nezávislost dvojice akcí a, b
 - Negativní efekty a mají prázdný průnik s předpoklady a pozitivními efekty b a naopak
 - Množina akcí je nezávislá při vzájemné nezávislosti všech dvojic akcí
- Vzájemné vyloučení (mutex) atomů nebo akcí
 - Např. jeden robot by byl na více místech současně či nakládal a vykládat současně
 - Řešení
 - Chtěli bychom zakázat podmnožiny atomů/akcí ve stavu - velmi drahé
 - Stačí zakázat dvojice - levné a uspokojivě přesné
 - Implementace
 - Vzájemně vyloučené akce: jsou závislé, nebo jsou jejich předpoklady mutex
 - Vzájemně vyloučené atomy
 - Akce, které mají každý z atomů jako pozitivní efekt, jsou mutex
 - Žádná akce nesmí mít jako pozitivní efekt oba atomy najednou
- Graphplan
 - Algoritmus střídající dvě fáze
 - Expanze plánovacího grafu, dokud v poslední vrstvě nejsou žádné dvojice atomů mutex
 - Extrakce paralelního plánu z plánovacího grafu tak, aby množina akcí byla bez mutexů
 - Tabulka nogoodů - zakázaných cílů pro detekci zacyklení
 - Pokud je extrakce neúspěšná, pokračuje se expanzí
 - Polynomiální časová a prostorová složitost
- Klasické přístupy
 - Dopředné plánování
 - Zpětné plánování: definuje inverzní akci γ^{-1}
 - Vylepšení s proměnnými (liftovaní): předchozí krok definujeme s proměnnými, místo abychom expandovali zpětně strom
 - Využití principu mgu (most general unifier)
 - Za proměnné se nakonec dosadí do hotového plánu
 - Lze je spojit pro omezení větvení, když se oba plány někde uprostřed potkají

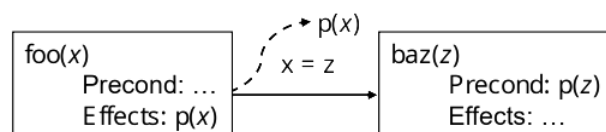
Kompilace plánování do jiných formalismů

- SAT
 - Atomy jsou výrokovými proměnnými
 - Pro každý časový krok, např. $at(r1, l1)^t \in \{True, False\}$
 - Akce jsou výrokové proměnné
 - Rovněž pro každý časový krok, např. $Move(r1, l1, l2)^t \in \{True, False\}$
 - Stavy jsou ohodnocení výrokových proměnných (atomů)
 - Některé jsou mutex - vyjádříme klauzulí, např. $\neg at(r1, l1) \vee \neg at(r1, l2)$, platí pak pro všechny časy t
 - Počáteční stav a cíl definujeme také klauzulí
 - Počáteční stav jako klauzule pro čas $t = 0$
 - Cíl v čase T , např. $at(r1, l2) \wedge at(r2, l1)$
 - Implementační detaily

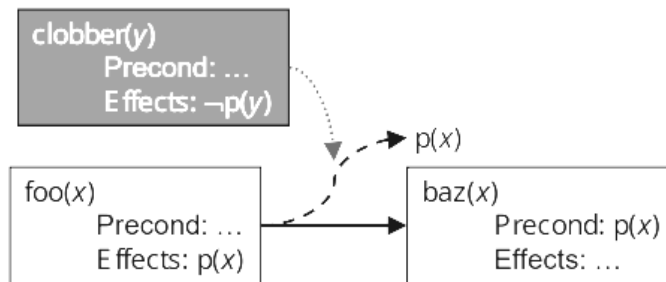
- Klausule pro změny pomocí akcí, např. $\neg p^t \wedge p^{t+1} \Rightarrow \bigvee_{p \in \text{effect}^+(a)} a^t$
- Klausule vynucující jediný krok v každém kroku
- SATPLAN
 - Využívá plánovací graf na určení atomů a akcí k reprezentaci pomocí proměnných
 - Zakóduje problém ve STRIPS jako CNF formuli, hledá řešení postupně v čase 1, 2, ... (časová expanze)
 - Splňování formule odpovídá extrakci plánu z plánovacího grafu
- CSP (Constraint Satisfaction Problem)
 - Atom je stavová proměnná, hodnoty odpovídají platnosti atomu s dosazenými konstantami, hodnota = stav v určitém čase
 - Stavové proměnné se detekují automaticky hledáním klik v grafu, kde vrcholy jsou výrokové proměnné a hrany jsou mutexy
 - Časová expanze jako v SAT
- Hierarchické plánování HTN (Hierarchical Task Networks)
 - Řeší problém příliš dlouhých plánů
 - Metody: sloučení operátorů vyjadřující komplexní akce
 - Název metody s parametry
 - Rozklad na jednodušší úkoly
 - Podmínky
 - Procedura postupně dekomponuje úkoly v úkolové síti
 - Díky dekompozici metod má lepší nasměrování na cíl než klasické plánování

Plánování v prostoru plánů

- PSP (Plan-Space Planning)
- Prohledáváme v prostoru částečných plánů
 - Počáteční plán obsahuje jen fiktivní počáteční a koncovou akci
 - Počáteční akce má efekt počátečního stavu, nemá předpoklady
 - Koncová akce má předpoklad cílového stavu, nemá efekty
- Postupné opravování částečného plánu
 - Otevřený cíl: v částečném plánu je akce s předpokladem $p(x)$, o kterém nevíme, jak jej nastavit



- Postup řešení
 1. Najdeme akci, která produkuje efekt p (ev. přidáme do plánu)
 2. Svážeme proměnné: nalezená akce má efekt $p(y)$, určíme $x = y$
 3. Nastavíme kauzální podmínku mezi akcemi
- Hrozba: existuje akce, která by mohla zrušit předpoklad pro nějakou akci

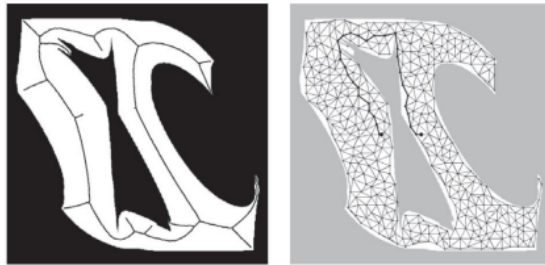


- Možnosti řešení
 - a) Přinutíme hrozbu, aby předcházela obě akce
 - b) Přinutíme hrozbu, aby následovala až po obou akcích
 - c) Svážeme proměnné nerovností, aby předpoklad nekolidoval
- Výsledkem je částečně uspořádaný plán, lze jej uspořádat podle kauzálních vazeb a získat sekvenci akcí vedoucí k cílovému stavu
- Složité na implementaci oprav

Plánování pohybu a problém lokalizace v robotice

- Typy pohybů
 - Z bodu do bodu: pohyb celého robota nebo jeho efektoru
 - Souladný: robot v kontaktu s překážkou
- Konfigurační prostor
 - Poloha, orientace, natočení kloubů
 - Hledání cesty mezi počáteční a cílovou konfigurací
 - Pohyb ve spojitém prostoru
 - Reprezentace
 - Pracovními souřadnicemi
 - Pozice prvků robota (kleští, zápěstí)
 - Vhodná na detekci kolizí - plně popisuje pozici
 - Ne všechny souřadnice jsou možné - zápěstí a kleště spojeny
 - Konfiguracemi
 - Natočení kloubů v úhlech
- Kinematika
 - Dopředná
 - Známe otočení kloubů, chceme určit polohu efektoru
 - Konfigurace (úhly) \rightarrow pracovní souřadnice
 - Inverzní
 - Známe pozici efektoru v pracovních souřadnicích, chceme určit konfiguraci
 - Složitá transformace, řešení rovnic s nekonečně mnoho řešeními (mnoho stupňů volnosti)
- Řešení spojitosti prostoru
 - Rozklad na buňky
 - Jednoduché na implementaci
 - Komplikuje se s rostoucí dimenzí
 - Problém nakládání s poloobsazenými buňkami
 - Skeletonizace
 - Převod hledání cesty na úlohy v jedné dimenzi

- Voroného diagram
 - Množina bodů se stejnou vzdáleností k překážkám
 - Tvoří graf, v němž je hledání cesty jednodimenzionální
 - Obtížná konstrukce
- Pravděpodobnostní mapa
 - Generujeme náhodné konfigurace, spojujeme ty, mezi nimiž vede snadná cesta
 - Součinnost s prohledáváním



- Dynamika a řízení
 - Kromě kinematického stavu je nutno uvažovat i rychlost (dynamický stav)
 - Určité zrychlení, hybnost
 - Plánování s dynamickými stavy je obtížné (diferenciální rovnice)
 - Metoda kontroleru
 - Stavovou proměnnou odchýlenou od požadavku robot kompenzuje působením efektoru
 - P (proporcionální kontroler): může oscilovat
 - PD (proporcionálně derivační): derivace působí jako tlumič, ale reaguje pomalu a na stálé odchylování přestane reagovat
 - PID (prop. derivačně integrační): systematické externí působení uvažováno v integračním faktoru
- Lokalizace
 - Přejímový model: známe počáteční polohu a model pro pohyb, deterministická stavová predikce (případně nejdřív globální lokalizace)
 - Senzorový model: skuteční roboti nejsou determinističtí, vztahujeme se k orientačnímu bodu detekovanému senzory
 - Polohu reprezentujeme jako Gaussovskou distribuci (nepřesnost měření)
 - Monte Carlo (particle filter): mračno vzorků odpovídajících stavům
 - Kalmanův filtr: přesvědčení reprezentujeme normálním rozdělením, transformujeme jej lineárními přechody

2 – Splňování omezení s konečnými doménami (CSP), pokročilé prohledávání (backjumping, dynamický backtracking), filtrace domén a lokální konzistenční techniky, globální omezení, rozhodovací heuristiky.

Splňování omezení s konečnými doménami (CSP)

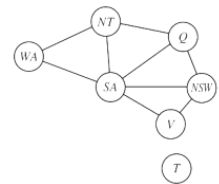
- CSP (Constraint Satisfaction Problem): trojice (X, D, C)
 - X : konečná množina proměnných, popisují vlastnosti řešení
 - D : konečná doména (obor hodnot) proměnných
 - C : množina podmínek nad X určující omezení pro rozhodnutí
- Příklady
 - Sudoku, proměnné $x_{i,j}$, podmínka *allDifferent*($x_{i,1}, \dots, x_{i,9}$)
 - CAD - parametrické modelování se splňováním omezení na bodech
- Rozdíly oproti prohledávání stavového prostoru
 - Stav má vnitřní strukturu danou proměnnými
 - Akce provádí lokální změnu ve stavu
 - Obecné prohledávací algoritmy
 - Obecné heuristiky
- Základní pojmy v řešení CSP
 - Stav: částečné přiřazení hodnot proměnným
 - Konzistentní: přiřazené hodnoty neporušují podmínky
 - Počáteční: žádná ohodnocení
 - Cílový: všechny proměnné konzistentně ohodnoceny
 - Akce: přiřazení hodnoty proměnné z její domény
- Základní algoritmus: chronologický backtracking
 - Odpovídá DFS, postupně přiřazujeme hodnoty
 - Pokud nelze přiřadit konzistentně, zkoušíme u poslední přiřazené proměnné jinou hodnotu
 - Neefektivní - čeká na ohodnocení všech proměnných v podmínce, konflikty zapomíná a objevuje znovu

Pokročilé prohledávání (backjumping, dynamický backtracking)

- Backjumping
 - Princip
 - Ohodnocujeme poslední proměnnou x , vzhledem k aktuálnímu přiřazení S' ohodnotit nelze
 - S' je konfliktní množina $conf_d = \{y_1, \dots, y_k\} \setminus \{x\}$ pro hodnotu $d \in D(x)$
 - Při rozšiřování ohodnocení do x vezmeme v úvahu sjednocení $Conf$ všech konfliktních množin $conf_{d_i}$ pro hodnoty $d_i \in D(x)$
 - Pokud nelze x ohodnotit, skok zpět na poslední ohodnocenou proměnnou z $Conf$
 - Při konfliktu místo skoku o jedno ohodnocení zpět tedy skáče dál
 - Konflikt navíc odhalíme dříve než při ohodnocení poslední proměnné
 - Algoritmus ale stále zapomíná odvedenou práci ve vedlejších větvích
- Dynamický backtracking
 - Pamatujeme si důvody (proměnné) nemožnosti ohodnotit všechny dosud navštívené proměnné
 - Při skoku se konzistentní část omezení ponechá, promažou se jen ty důvody, jejichž ohodnocení jsme změnili

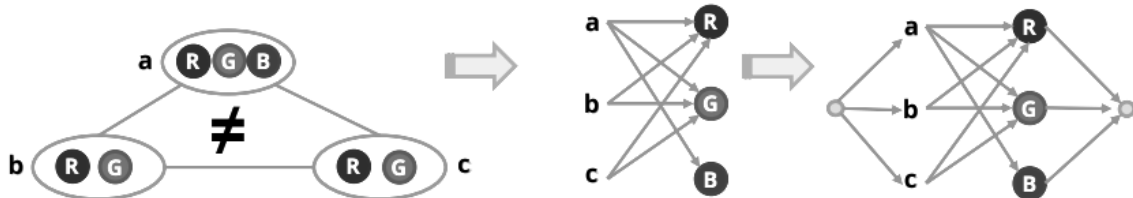
Filtrace domén, lokální konzistenční techniky

- Filtrace domén
 - Dopředná kontrola podmínek
 - Pracovní domény pro proměnné
 - Při ohodnocení proměnné z domény vyškrtneme hodnoty odporující podmínkám
 - Chyba při ohodnocování nastane, když je pracovní doména prázdná
- Důraznější filtrace
 - Proškrtaváme domény i dosud neohodnocených proměnných
- Hranová konzistence
 - Filtrace pro CSP s binárními podmínkami, chápeme jako graf
 - Pro každou podmínku $c \in C$ nad proměnnými x_1, x_2 kontrolujeme orientovanou hranu (x_1, x_2)
 - Pro každou hodnotu z domény $d_1 \in D$ hledáme podporu $d_2 \in D$ vzhledem k c
 - Pokud d_1 nemá podporu v $D(x_2)$, odebereme d_1 z $D(x_1)$
 - Takto filtrujeme domény i dosud neohodnocených proměnných



Globální omezení

- Zobecněná hranová konzistence
 - Zobecnění na n-ární podmínky
 - Hledáme podporu mezi všemi proměnnými v rámci jedné podmínky
 - Místo testování k-tic hodnot modelujeme jako hledání párování
 - Maximální tok v jednotkové síti
 - Vyřazení neoznačených hran promítneme do pracovních domén



- Rozbívání symetrie
 - Např. u problému N královen
 - Přidáváme podmínku pro lexikografické uspořádání hodnot proměnných

Rozhodovací heuristiky

- Výběr proměnné
 - Výběr nejvíce omezené proměnné (nejméně hodnot v doméně)
 - Úzké hrdlo problému, těžké rozhodnutí
 - Nechceme odkládat těžká rozhodnutí
 - Výběr “klíčové” proměnné (učastní se nejvíce podmínek)
 - Nechceme odkládat důležitá rozhodnutí
- Výběr proměnné
 - Hodnota, která nejméně omezí dosud neohodnocené proměnné
 - Ponechá větší volnost pro pozdější proměnné
- Tyto heuristiky jsou nezávislé na problému
- Lze využít i strukturu problému (samostatné řešení komponent souvislosti v grafu)

3 – Systematické a lokální splňování v logice (DPLL, CDCL, WalkSAT, posílání zpráv).

Automatické uvažování, rozhodování v teoriích prvního řádu, obecná rezoluce, princip SAT-modulovaných teorií (SMT). Zpracování přirozeného jazyka.

- Motivace
 - Mnoho praktických úloh je NP
 - Všechny lze převést na SAT, pro nějž existuje mnoho efektivních řešičů
- Problémy
 - Absence aritmetiky ve výrokovém světě
- Příklady úloh
 - Formální verifikace HW/SW
 - Bioinformatika, genetika - konzistence dědičnosti
 - Plánování - hledání plánu na n kroků
- Pojmy z logiky
 - Literál: proměnná nebo její negace
 - Klausule: disjunkce literálů
 - Term: konjunkce literálů
 - CNF se podobá CSP - klauzule jsou podmínky, formule je konjunkce klauzulí
- Systematické metody poskytují garance
 - Úplnost: vždy skončí a odpoví narozdíl od genetických apod.
 - Vysvětlitelnost: poskytují formální zdůvodnění odpovědi - explainable AI
 - Pro speciální třídy formulí polynomiální čas (např. 2-SAT)
- Automatické uvažování
 - Automatické dokazování vět, ověřování důkazů
 - Uvažování s neurčitostí
 - ATP: algoritmy dokazující, že zadané matematické tvrzení je logickým důsledkem daných axiomů
 - APC: interaktivní verze ATP, uživatel zadává důkazové kroky a algoritmus je verifikuje vzhledem k axiomům a předchozím důkazům

Systematické splňování v logice (DPLL, CDCL)

- DPLL
 - Backtracking
 - Propagace čistých proměnných: vyskytují se pouze jako pozitivní či pouze jako negativní literál, lze je hned ohodnotit
 - Jednotková propagace
 - Jednotková klauzule se ohodnotí *True*
 - Následně ohodnocujeme klauzule, kde zbývá jen jeden neohodnocený literál

- Implikační graf (orientovaný, vrstevnatý) - zachycení konfliktu
 - Ukládá předchůdcovské klauzule při jednotkové propagaci
 - Vrcholy jsou ohodnocení proměnných podle hladiny
 - Hrany jsou klauzule zapřičiňující ohodnocení proměnné, kam vedou
 - Speciální vrchol konfliktu má také důvodovou klauzuli v hraně
 - Hranový řez v implikačním grafu
 - Umožňuje extrahovat konfliktní klauzuli, která konflikt zakazuje
 - Ta je přidána mezi ostatní klauzule a konflikt je naučen
- CDCL (Conflict Driven Clause Learning)
 - Decide: ohodnotí neohodnocenou proměnnou
 - Jednotková propagace
 - Přitom se staví implikační graf
 - Konflikt
 - Vytvoří z implikačního grafu vhodnou konfliktní klauzuli (nogood)
 - Potřebujeme tzv. vynucující konfliktní klauzuli
 - Z poslední vrstvy obsahuje pouze právě ohodnocovanou proměnnou
 - Tím je dosaženo větvení, které CDCL explicitně nemá
 - Krátká konfliktní klauzule je využitelná pro jednotkovou propagaci při opravě konfliktu, také je obecnější
 - Konfliktní klauzule se využívá k zpětnému skoku - backtrack (ekvivalent backjumpingu v CSP)
 - Vynucující konfliktní klauzuli lze po využití k backtracku zapomenout
- Heuristiky pro výběr literálu
 - VSIDS (Variable State Independent Decaying Sum)
 - Každý literál má skóre, to se inkrementuje naučením klauzule, která jej obsahuje
 - Periodicky skóre dělíme konstantou $d > 1$
 - Literál s nejvyšším skóre ohodnocujeme *True*
 - Preferuje se tak splňování nových konfliktů před původními klauzulemi
 - Berkmin
 - Proměnné a literály mají VSIDS skóre, dělí se jen u proměnných
 - Konfliktní klauzule se ukládají na zásobník
 - Nerozhodnutá klauzule se bere ze zásobníku
 - V ní proměnná podle skóre, polarita podle skóre literálů
 - Při prázdném zásobníku vybíráme přímo proměnné podle skóre
- Efektivní jednotková propagace
 - Pro rychlé určení jednotkové klauzule bez procházení všech
 - Dva sledované literály v každé klauzuli
 - Pro každou proměnnou ukládáme sledované výskyty
 - Postup
 - Ohodnotíme jeden sledovaný literál
 - Hledáme nový ke sledování místo právě ohodnoceného
 - Pokud náhradní nelze nalézt, klauzule je jednotková
 - Při backtrackingu není třeba nic dělat, sledování zůstává platné

Lokální splňování v logice (WalkSAT, posílání zpráv)

- GSAT
 - Hladové procházení úplného ohodnocení - neúplný algoritmus
 - Postupně zkoušíme negovat ohodnocení jednotlivých proměnných
 - Účelová funkce: Počet splněných klauzulí
 - Heuristika často skončí v lokálním maximu, proto několik náhodných restartů
- WalkSAT
 - Náhodná procházka, procházení úplných ohodnocení s restarty
 - Střídání hladového kroku s pravděpodobností p a náhodného kroku s pravděpodobností $p - 1$
 - Umožňuje uniknout z lokálního maxima
- Algoritmy posílání zpráv
 - Klauzule jako paralelní funkční jednotky
 - Zkoumá, jak změna hodnoty proměnné ovlivní sousední klauzule
 - Propagace výstrah: klauzule posílají proměnným zprávy (výstrahy), jak chtějí nastavit proměnnou, aby byla splněna
 - Výstrahami inspirovaná decimace: postupně zjednodušuje formuli dosazováním

Rozhodování v teoriích prvního řádu

- Logika prvního řádu
 - Jazyk: proměnné, spojky, kvantifikátory, pomocné symboly
 - Signatura (nelogické symboly): funkce a predikáty
 - Formule
 - Term: proměnná nebo funkce nad termy
 - Atom: predikátový symbol nad termem
 - Formule: atom, rekurzivně pak formule spojená spojkami či s aplikovaným kvantifikátorem
 - Teorie: množina formulí (i nekonečná)
 - Důkaz formule ϕ z teorie T : posloupnost formulí končící ϕ , kdy každá formule patří do T nebo je z nich odvozena
 - Pravdivost ϕ v teorii T
 - Struktura signatury: realizace funkcí či predikátů nad modelem (strukturou splňující formule teorie - sporná teorie nemá model)
- V podstatě veškerá informatika lze formalizovat pomocí logiky prvního řádu - silný vyjadřovací prostředek
- Neexistuje algoritmus, který by rozhodoval platnost tvrzení v logice prvního řádu
 - Máme jen semi-rozhodnutelnost nebo rozhodnutelnost ve speciálních případech (fragments)

Obecná rezoluce

- Semi-rozhodování (rezoluce) nad logikou prvního řádu
 - Rezoluční metoda
 - Dokazujeme tvrzení vzhledem množině axiomů v teorii
 - Sjednocujeme teorii s negací tvrzení a snažíme se odvodit spor

- Rezoluční pravidlo
 - Výroková varianta (řez)

$$\frac{(p \vee A) \wedge (\neg p \vee B)}{A \vee B}, \text{ kde } A \text{ a } B \text{ jsou disjunkce literálů. } A \vee B \text{ se nazývá resolventou.}$$

- Obecná varianta pro logiku prvního řádu

$$\frac{(p \vee A) \wedge (\neg q \vee B)}{(A \vee B)\sigma}, \text{ kde } \sigma \text{ je substituce unifikující } p \text{ a } q.$$

- Unifikuje se více pozitivních literálů v jedné a negativních v druhé klauzuli - nejobecnější unifikace
- Příprava na rozhodování
 - Formule teorie a tvrzení převedeme na CNF
 - Standardizujeme proměnné: pro odstranění kvantifikátoru je nutné přejmenovat proměnnou
 - Skolemizace: odstraňujeme existenční kvantifikátory
 - Výraz $\exists x (P(x))$ nahradíme $P(A)$, kde A je nová konstanta
 $\forall x [Animal(A) \wedge \neg Loves(x, A)] \vee Loves(B, x)$
 - Náhrada konstant za Skolemovy funkce závisících na vnější kvantifikované proměnné
 $\forall x [Animal(F(x)) \wedge \neg Loves(x, F(x))] \vee Loves(G(x), x)$
 - Zahodíme všeobecné kvantifikátory
 $[Animal(F(x)) \wedge \neg Loves(x, F(x))] \vee Loves(G(x), x)$
 - Distribuce pomocí \wedge, \vee
 $[Animal(F(x)) \vee Loves(G(x), x)] \wedge [\neg Loves(x, F(x)) \vee Loves(G(x), x)]$
 - Eliminujeme spojky jiné než \neg, \wedge, \vee
 - Negace převedeme až k literálům (s prohazováním kvantifikátorů)
- ANL loop: algoritmus pro rezoluční metodu

Princip SAT-modulovaných teorií (SMT)

- Spolupráce SAT solveru a $DECIDE_T$ rozhodování teorií zabudovaná do CDCL
- SAT solver
 - Vybírá, které literály splnit, aby byla splněna Booleovská struktura
 - Nerozumí teorii T
- $DECIDE_T$
 - Kontroluje, jestli literály vybrané SAT solverem mohou současně platit
 - Pokud ne, hlásí zpět SAT solveru formou zakázaného ohodnocení (konfliktní klauzule v CDCL)

Zpracování přirozeného jazyka

- NLP (Natural Language Processing)
- V multiagentním světě (roboti a lidé) lze komunikací najít shodu na plánech
- Formální jazyk
 - Gramatika $G = (N, T, S, P)$

- N: konečná množina neterminálů
- T: konečná množina terminálů
- $S \in N$: počáteční neterminál
- P: konečná množina odvozovacích pravidel
- Chomského hierarchie gramatik
 - Rekurzivně spočetné
 - Kontextové: pravidla tvaru $\alpha X \beta \rightarrow \alpha w \beta$, kde w je neprázdný
 - Bezkontextové: pravidla tvaru $X \rightarrow w$
 - Regulární: pravidla tvaru $X \rightarrow x$ nebo $X \rightarrow xY$
- Fáze komunikace
 - Záměr sdělit skutečnost
 - Generování přesvědčivého sdělení o skutečnosti -> věta
 - Syntéza: sdělení, hlasový syntezátor -> zvuk
 - Vnímání: rozpoznání hlasu -> věta
 - Analýza: věta -> derivační strom
 - Odstranění dvojsmyslů: pravděpodobnostní interpretace
 - Začlenění: zda věřit sdělení či nikoliv
- PCFG (Probabilistic Context-Free Grammar)
 - Přirozený jazyk lze modelovat bezkontextovou gramatikou
 - Každé pravidlo má přiřazenou pravděpodobnost
 - Derivační strom (odvození) má pak také pravděpodobnost
 - Učení pomocí korpusu korektních derivačních stromů vzhledem k pravidlům
- CYK algoritmus: parsování pomocí PCFG v Chomského normální formě
 - Pravidla tvaru $X \rightarrow YZ$, nebo $X \rightarrow w$
- Lexikalizované PCFG
 - Pravděpodobnost pravidel závisí na vtahu slov v derivačním stromu, nikoliv jen na sousedství ve větě
 - Uvažujeme nejvýznamnější slovo ve spojení - pravděpodobnost všech dvojic by bylo náročné počítat
 - Stále je to ale mnoho dvojic slov, pro která potřebujeme znát pst.
- DCG (Definite Clause Grammars)
 - Gramatiky s určenou klauzulí - navíc používáme podmínku
 - Pravidlo platí, pokud je podmínka splněná
 - `Compatible("black", "dog")` je splněno, ale `Compatible("black", "sugar")` nikoliv
 - Silný nástroj pro řešení skloňování a pádů
 - Podmínky zamezí nutnosti dělit pravidla na podtypy např. podle shody podmětu s přísudkem a časování
- Sémantika
 - Po syntaktické analýze je nutné pochopit význam věty
 - Chceme získat interpretaci, která není term ani atom, využijeme lambda notaci pro dosažení atomů
 - Problém s kvantifikací: *Every agent feels a breeze.* má více významů
- Strojový překlad
 - Statistický překlad
 - Učení pravděpodobností PCFG - máme jen gramatiku a korpus, stromy chybí
 - Text rozdělíme na slovní spojení, hledáme spojení v druhém jazyce s maximální pravděpodobností

4 – Metody pro hodnocení a výběr příznaků
(univariетní/multivariетní metody,
filtrační/wrapper/embedded metody).
Selektivní/adaptivní metody redukce počtu
instancí: Condensed Nearest Neighbor (CNN),
Delauney/Gabriel/RNG grafy, Wilsonova editace,
Multi-edit metoda, Tomkovy spoje.

Metody pro hodnocení a výběr příznaků

- Chceme vybrat příznaky, které nesou nejvíc informace, nejlépe oddělují data
- Zjednodušení pro snížení dimenzionality a zrychlení modelu
- Odpověď na “jak moc je příznak X_i relevantní pro prediky Y ?”

Univariетní/multivariетní metody

- Univariетní metody
 - Zvažují každý příznak nezávisle
 - Následující zjišťují závislost mezi jedním X_i a Y
 - T-test
 - Párový, dvouvýběrový
 - Ověření hypotézy o rovnosti střední hodnoty
 - Korelace
 - Lineární vztah mezi proměnnými
 - Pearsonův korelační koeficient
 - Kovariance normovaná směrodatnými odchylkami
 - Hodnoty mezi -1 a 1
 - Korelační a kovarianční matice
 - Spearmanův korelační koeficient
 - Entropie
 - Očekávaná míra vlastní informace v náhodné proměnné
 - Vzájemná informace
 - Podíl sdružené pravděpodobnosti a marginálních pravděpodobností
 - Kolik informace jedné proměnné je vysvětleno druhou
- Multivariетní metody
 - Zvažují podmnožiny příznaků najednou

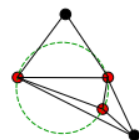
Filtrační/wrapper/embedded metody

- Filtrační metody

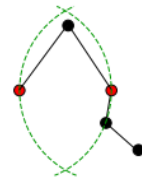
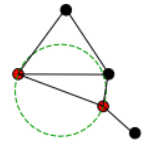
- Seřazení příznaků nebo jejich podmnožin podle relevance nezávisle na klasifikátoru
- Pro seřazení příznaků např. korelace, entropie, ...
- Hodnocení výsledku statistickými testy
- Odolné proti přeučení, ale nemusí najít vše
- Wrapper metody
 - Hledání nejlepší podmnožiny příznaků podle výstupu klasifikátoru
 - Lze využít cross-validaci, poté zprůměrovat kvalitu prediktorů
 - Najde nejužitečnější příznaky, ale pozor na přeučení
- Embedded metody
 - Stejně jako wrapper, ale hledání je řízeno přímo klasifikátorem
 - Také lze využít cross-validaci
 - Menší riziko přeučení a menší výpočetní náročnosti

Selektivní/adaptivní metody redukce počtu instancí

- Nearest Neighbor problem
 - Pro získání nejbližších sousedů je třeba získat vzdálenost ke všem
 - To je výpočetně a paměťově náročné, navíc prokletí dimenzionality
 - Chceme proto redukovat počet dat, ale aby to neublížilo predikci
 - Můžeme také zjednodušit kNN na 1NN přepsáním některých labelů
- Condensing
 - Zanechávání pouze prvků nutných k vytvoření decision boundary
 - Konzistentní podmnožina: vytváří stejné hranice
 - Minimální podmnožina: nejmenší taková, že klasifikuje původní data stejně jako úplná
 - CNN (Condensed Nearest Neighbor)
 - Inkrementální, závisí na pořadí bodů
 - Neprodukuje konzistentní ani minimální podmnožinu
 - $O(n^3)$ brute-force
 - Začne s jediným vzorkem
 - Postupně přidává špatně klasifikované vzorky po jednom (tady to právě závisí na pořadí)
 - Dokud nejsou špatně klasifikované, nebo všechny přidané
- Proximity graphs
 - Opět zanechávání prvků pro decision boundary
 - Na principu hledání sousedství prvků z opačné třídy
 - Delaunay
 - Delaunay triangulace: tři body jsou sousedy, pokud uvnitř kružnice jimi proložené neleží žádný jiný bod (opsaná trojúhelníku)
 - Voronoi diagram: obarvení plochy tak, že oblast kolem každého bodu obsahuje body nejbliž právě jemu
 - Dualita diagramu a triangulace: propojením středů kružnic získáme hranice Voroného diagramu
 - Ponecháme body, jejichž sousedé jsou z opačné třídy
 - Produkuje konzistentní hranici
 - Drahý výpočet
 - Ponechává body navíc



- Gabriel
 - Podmnožina Delaunay
 - Dva body jsou sousedé, pokud kružnice jimi proložená (nad průměrem) je prázdná
 - Nedává konzistentní hranici, ale je levnější
- RNG (Relative Neighborhood Graph)
 - Podmnožina Gabriel
 - Dva body jsou sousedé, pokud "luna" jimi proložená je prázdná (vzdálenost je poloměr, středy v obou bodech)
 - Nedává konzistentní hranici, ale ještě více redukuje
- Editing
 - Princip odebrání šumových vzorků pro získání hladší hranice
 - Výsledkem jsou homogenní shluky
 - Wilson
 - Odebrání bodů, které se neshodují s třídou většiny z k sousedů
 - Multi-edit
 - Opakovaná aplikace Wilsona na náhodné části, klasifikace pomocí 1NN
 - Algoritmus, opakuje se, dokud jsou změny
 - Rozdělení na náhodné podmnožiny
 - Klasifikace pomocí 1NN
 - Editing: odebrání špatně klasifikovaných vzorků
- Tomek links
 - Tomkův spoj: spoj bodů odlišné třídy, které nemají jiný bližší bod
 - Odebíráme prvky majoritní množiny, které jsou součástí spojů
 - Odebereme tak šum a hraniční případy
 - Výpočetně náročné
 - Samy o sobě neúčinné, dobré ale v kombinaci s jinou metodou



5 – Algoritmy pro nahrazování chybějících hodnot. Detekce a ošetření odlehlých hodnot. Vyvažování skupin dat (undersampling/oversampling metody).

Nahrazování chybějících hodnot

- Ve zdrojových datech nemusíme být schopni odlišit chybějící a schválně nevyplněnou hodnotu
- Mechanismy chybějících dat
 - Missing completely at random: příznak chybí nezávisle na své hodnotě a hodnotách ostatních příznaků v řádku
 - Missing at random: nepřítomnost může záviset na hodnotách ostatních příznaků
 - Missing not at random: nepřítomnost může záviset na hodnotě
- Způsoby nahrazení chybějících hodnot
 - Nedělat nic
 - Použije se relevantní reprezentace tak, aby šel aplikovat model
 - Např. NaN = -1, pokud všechny ostatní hodnoty jsou kladné
 - Vymazání
 - Listwise: smaže celý řádek, kde chybí alespoň jedno pole
 - Pairwise: jako listwise, ale zaměřeno na důležité příznaky
 - Příznaku: pokud chybí pro většinu datových bodů
 - Imputace
 - Doplnění chybějících hodnot nějakým způsobem, nejlepší není
 - Průměr či medián sloupce
 - Průměr kNN - vhodné pouze pro nízkodimenzionální data

Detekce a ošetření odlehlých hodnot

- Odlehlé hodnoty: anomálie odchylující se od ostatních pozorování
- Detekce
 - Nesupervizované metody, hlavně shlukování
 - Univariální pohled
 - Rule of thumb: hodnota mimo interval založený na rozptylu ($Q_1 - 1,5 * IQR$; $Q_3 + 1,5 * IQR$), kde Q_i je i -tý kvartil, IQR je prostředních 50%
 - Ručně stanovená hranice
 - Multivariální pohled
 - Shlukování založené na vzdálenosti (K-means, LOF)
 - Density-based přístupy (SVM)
 - Modelové přístupy (autoencodery, LSTM)
- Po identifikaci je nutná analýza původu odchylky (lidská chyba, měření, experimentální, samplingová...)

- K-means
 - Iterativně tvoříme k center
 - Přiřazujeme prvky nejbližším centrům
 - Přepočítáme centra jako průměry přiřazených prvků
 - Myšlenka, že každý bod shluku by měl být blízko jeho středu
- LOF (local outlier factor)
 - Hledání anomálnosti měřením lokální odchylky bodu vzhledem jeho sousedům
 - $LOF(a)$ = průměrná lokální hustota sousedů dělená lokální hustotou a
 - Kolem 1: a je podobný sousedům
 - Nad 1: a je v řidší oblasti než sousedé - indikuje anomalitu
 - Výhoda lokality a dobré geometrické interpretace pro nízkodim. Prostor
 - Nevýhoda těžké interpretovatelnosti výsledků

Vyvažování skupin dat

- Nevybalancovaná data
 - Většina dat patří do jediné třídy
 - Chceme vybalancovat poměr tříd pro zlepšení predikce
 - Problém falešně dobrého TPR apod.
- Undersampling
 - Random undersampling
 - Náhodná eliminace příkladů z majoritní třídy
 - Může zničit potenciálně důležitá data
 - Tomek links
 - Tomek link: dva prvky opačné třídy, které nemají bližšího souseda
 - Odstranění majoritního prvku, který je v Tomkově spoji
 - CNN (Condensed Nearest Neighbor)
 - Výběr bodů tvořících hranici mezi třídami
 - Neprodukuje konzistentní ani minimální podmnožinu (netvoří identickou rozhodovací hranici ani neodebere maximum bodů)
 - $O(n^3)$ brute-force
 - Začne s jediným vzorkem
 - Postupně přidává špatně klasifikované vzorky po jednom (kvůli tomu metoda závisí na pořadí bodů)
 - Dokud nejsou špatně klasifikované, nebo všechny přidané
 - Tomek + CNN: nevhodné, hledání Tomkových spojů je náročné
 - CNN + Tomek
 - NCL (Neighborhood Cleaning Rule)
 - Odebírá majoritní prvky hlavně pro zčištění dat než pro redukci
 - Odebere majoritní, pokud 3 nejbližší sousedí jsou minoritní
 - Odebere 3 majoritní, pokud jsou nejbližšími sousedy minoritního
 - ENN (Extended Nearest Neighbor)
 - Odebere každý bod, jehož třída se liší od třídy aspoň dvou ze tří jeho nejbližších sousedů
 - Agresivnější odebírání než Tomek

- Oversampling
 - Random oversampling
 - Náhodná replikace příkladů z minoritní třídy
 - Zhoršuje overfitting
 - SMOTE (Synthetic Minority Oversampling Technique)
 - Tvorba minoritních vzorků interpolací jiných blízkých
 - Náhodně na úsečce spojující minoritní vzorek s jeho k sousedy
 - Při výskytu outlierů generuje šum
- Kombinace under- a oversamplingu
 - SMOTE + Tomek
 - SMOTE může vytvořit umělou minoritu někde v majoritní části
 - Následné odebrání obou vzorků tvořících Tomkův spoj
 - SMOTE + ENN
 - Pročistí více než Tomek
- Oversampling bývá lepší než undersampling (nepřijdeme o informaci), ale kombinace může být ještě lepší

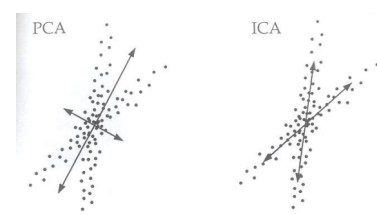
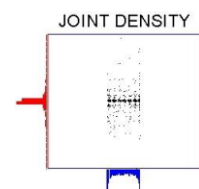
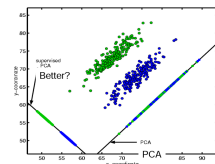
6 – Lineární projekce dat do prostoru o méně dimenzích: metoda hlavních komponent (PCA), lineární diskriminační analýza (LDA). Nelineární metody redukce dimensionality (Sammonova projekce).

Projekce dat do prostoru o méně dimenzích

- Snížení dimenze při zachování maxima informace (vzájemných vztahů bodů)
- Vyřazení redundantní informace, zrychlení výpočtu
- Nová podoba dat ale může být těžko interpretovatelná
- Každá metoda se snaží minimalizovat nějaké kritérium

Lineární projekce

- Náhodná projekce
 - Pomocí náhodné matice se sloupce jednotkové délky
 - Johnson-Lindenstrauss: při náhodné projekci se vzdálenosti mezi body zhruba zachovávají
 - Není to úplně projekce, protože není nutně ortogonální
- PCA (Principal Component Analysis)
 - Redukce dimensionality za zachování maximálního rozptylu dat
 - Minimalizace information loss: $\min \|x - x'\|$
 - Cílový prostor
 - Střed v průměru vzorků
 - Směry dané hlavními komponentami: vlastní vektory největších vlastních čísel kovarianční matice
 - Kovarianční matice je reálná symetrická, proto jsou hlavní komponenty ortogonální a tvoří množinu báze vektorů
 - Nemusí nalézt optimální vzor v datech a podle něj je oddělit - nemá totiž znalost o labelu, příslušnosti dat k třídě
 - Vhodnější pro menší data
- ICA
 - Metoda rozdělování signálů lineární transformací
 - Součet rozdělení je víc Gaussovský, takže chceme oddělit nejvýše jeden Gaussovský signál
 - Rotace sdruženého signálu tak, že maximalizujeme rozptyl jednoho z nich
 - Extrahuje v neurčeném pořadí a polaritě
 - Vhodné při slepém oddělování dat bez znalosti jejich významu



- LDA (Linear Discriminant Analysis, Fisherova projekce)
 - Redukce dimenzionality za zachování co nejvíce diskriminační informace
 - Hledá směry, ve kterých jsou nejlépe separovány známé třídy
 - Maximalizace vzdáleností průměrů tříd (scatter matice S_w)
 - Minimalizace rozptylu uvnitř tříd (scatter matice S_b)
 - Pro C tříd hledáme projekci U , která maximalizuje podíl S'_b/S'_w transformovaných dat
 - Projekční matici získáme složením vlastních vektorů z řešení obecného problému vlastních vektorů do sloupců U

$$S_b U_k = \lambda_k S_w U_k$$
 - Protože S_b má hodnot nejvýše $C - 1$, projekce je do prostoru dimenze nejvýše $C - 1$
 - S_w může být singulární, pak nejprve aplikujeme PCA
 - Vhodné při velkém, reprezentativním balíku dat

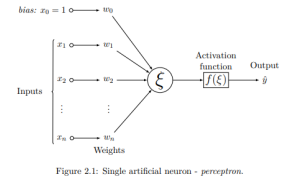
Nelineární projekce

- Sammonova projekce
 - Mapování vysokodimenzionálního prostoru na nižší tak, že se co nejlépe zachovávají vzdálenosti mezi body
 - Netransformuje souřadnice, ale reorganizuje pozice bodů
 - Minimalizace Sammonovy chyby např. gradientním sestupem
 - Normalizovaný průměrný kvadrát rozdílu vzdáleností každých dvou bodů v původním a novém prostoru

7 – Učení dopředných neuronových sítí, konvoluční neuronové sítě a jejich regularizace.

Dopředné neuronové sítě

- Perceptron
 - Nejjednodušší binární klasifikátor
 - Lineární kombinace
 - Bias pro posun
 - Nelineární aktivační funkce
 - Perceptronový algoritmus učení
 - Pouze pro lineárně separabilní problémy
 - Volíme úvodní vektor vah jako rozdíl průměrů kladných a záporných vektorů
 - Opakovaně vybíráme náhodný vektor z trénovacích dat
 - Pokud je z třídy 1 a klasifikován jako 0, přičteme ho k vektoru vah
 - Pokud je z třídy 0 a klasifikován jako 1, odečteme
 - Gradientní učení
 - Konverguje i pro lineárně neseperabilní data
 - Optimalizace ztrátové funkce (např. SSE)
 - Změna vah ve směru proti gradientu škálovaném learning rate
 - Potřebujeme parciální derivaci ztrátové funkce vůči každé váze
 - Cross-entropy loss (log loss)
 - Ztrátové funkce pro klasifikaci do c tříd
 - $L(Y, p) = -\log(p_y)$
 - Velká hodnota pro hodně jisté, ale špatné predikce
 - Když je prvek ve třídě 1, ale je ho příslušnost predikujeme s pst. blízko 0
- MLP (Multi Layer Perceptron)
 - Několik fully-connected vrstev
 - Oddělí i nelineárně separabilní data
 - Třívrstvá síť dokáže modelovat jakoukoli spojitou funkci
 - Využití v supervizovaném učení
 - Pro gradientní sestup potřebujeme parciální derivace
 - Gradient složené funkce získáme řetězovým pravidlem
 - Stačí hodnoty pronásobit
 - Backpropagation
 - Dopředným chodem získáme výstup a jeho ztrátovou funkci
 - Zpětně od poslední vrstvy rekurzivně počítáme gradienty
 - Upravujeme váhy podle learning rate
 - Dokud není splněno ukončovací kritérium (delta, threshold)
 - Moment setrvačnosti
 - Při updatu vah přičítáme i část gradientu z předchozího kroku
 - Urychlení konvergence, únik lokálnímu optimu, regulace přehnané reakce na vzorek



$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial x}$$

- Batch updates: update vah až pro kumulovanou chybu více vzorků
- Adaptive learning rate: pro každý neuron, může se měnit během učení
- Adaptive slope: parametrizovaná aktivační funkce, lze také učit
- Metody druhého řádu: použití druhé derivace nebo její aproximace
- ADAM (Adaptive Moment Estimation)
 - Moment setrvačnosti: exponenciálně odezdnávající moving average gradient
 - Adaptivní learning rate pro jednotlivé váhy
 - Penalizace vah způsobujících velké oscilace
 - Zamezení overfittingu
- Learning to optimize: optimalizační algoritmus je naučen druhou sítí
- SOM (Self-Organizing Maps)
 - Nesupervizovaná neuronová síť pro redukci dimenzionality
 - Neighborhood funkce: snaha zachovat topologické vlastnosti prostoru
 - Váhy neuronů popisují oblast prostoru

Konvoluční neuronové sítě

- Pro situace, kde by měl MLP příliš mnoho vah (typicky obrázky)
 - Redukce vah jejich sdílením v konvolučních filtrech (kernelech)
 - Ručně vytvořené kernely
 - Gaussovský filtr: rozostření
 - LoG edge detector (Laplacián Gausiánu): vyhladí Gaussem, detekuje hrany (se zdvojením) Laplaceovským filtrem
 - Typy vrstev
 - Fully connected
 - Konvoluční
 - Neurony v následující vrstvě jsou spojeny jen s "okolními" z předchozí vrstvy
 - Váhy neuronů jsou stejné (stejný kernel)
 - Detekce přítomnosti lokální vlastnosti na celém prostoru
 - Padding: pro okraje konvoluce
 - Stride: délka kroku při přechodu konvoluce
 - Pooling
 - Redukce dimenzionality
 - Následuje konvoluční vrstvu
 - Max pooling, average pooling
- (a) Average pooling operation.

(b) Max pooling operation.
- Konvoluce extrahuje nejprve obecné příznaky (hrany, rohy, křivky), poté textury a komplikovanější vzory
 - ConvNet, AlexNet, GoogLeNet, ResNet (s přeskoky mezi konvolucemi)

Regularizace

- Dropout
 - Náhodné nulování výstupu některých neuronů
 - Síť se tak nenaučí spoléhat pouze na pár důležitých neuronů
- L1/L2: penalizace navíc absolutní hodnotou/čtvercem vah - sparsity/difussion

8 – Autoencodery a generativní neuronové sítě.

Autoencodery

- Dopředné sítě, které predikují vstup
- Uprostřed se nachází úzké hrdlo - zakódovaný/embedded vstup
- Využití
 - Redukce dimenzionality, komprese
 - Extrakce abstraktních příznaků pro využití v supervizovaném modelu
 - Detekce anomálií (reconstruction error)
- Stacked AE
 - Řada postupně zmenšujících se autoenkodérů
 - Po natrénování jednoho se zahodí dekompresní vrstva a připojí nový
 - U poslední vrstvy vstup do klasifikátoru/prediktoru se supervizí
 - Každá vrstva hladově extrahuje nejlepší příznaky, tudíž jsou smysluplné
- Porovnání s PCA
 - Výhody AE
 - AE umí modelovat nelineární vztahy
 - PCA příznaky jsou ortogonální, tudíž nekorelované - u AE to nemusí platit
 - Výhody PCA
 - PCA je výpočetně levnější a rychlejší
 - AE je náchylnější na overfitting kvůli většímu počtu parametrů
- Regularizace
 - Dropout
 - L1 regularizace: penalizace vah v loss funkci, vede na řidší váhy
 - Sparsity regularizace: penalizace počtu aktivních latentních neuronů
 - Denoising AE: zašumění vstupu, aby se AE naučil generalizovat a ne jen mapovat identitu

Generativní neuronové sítě

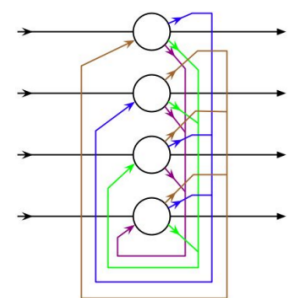
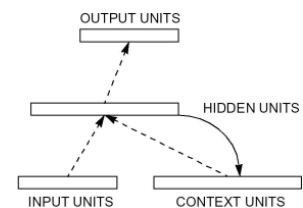
- Snaha modelovat distribuci vstupu pro pozdější generování dat ze stejné distribuce
 - Generování obrázků a textu
 - Generování dat pro učení, balancování datasetů...
- GMM (Gaussian Mixture Models)
 - Modelování distribuce jako váženého součtu Gaussovských distribucí
 - EM (Expectation Maximization) algoritmy
 - Princip náhodné inicializace, pak upřesňování (např. K-means)
 - Pro GMM funguje jako MLE odhad parametrů rozdělení
 - Využití autoencoderu
 - Klasické použití autoenkodéru a fitting GMM na embedded space
 - Embedding nemusí prostor zmenšit dostatečně
 - Tendence k overfittingu
 - Omezení latentního prostoru na parametry Gaussovské distribuce
 - Dekódovací část pak generuje vzorek z této distribuce
- Autoregresivní generativní modely

- Predikce časových řad na základě předchozího chování
- Data musí být autokorelovaná - využijeme znalost postupných změn
- PixelRNN
 - Predikce pixelů jako časových řad
 - Row LSTM: jednosměrné (od rohu)
 - Bidirectional LSTM: z obou konců zároveň
 - Díky paměti lze hledat i netriviální sekvence
 - Pomalé, nelze paralelizovat
- PixelCNN
 - Výběr dopočítávaných částí obrázku pomocí konvoluce
 - Maskovaná konvoluce pro vynucení autoregrese
 - Super resolution: postupné generování struktury a poté doplňování detailů
 - Lepší paralelizovatelnost
- GAN (Generative Adversarial Networks)
 - Generátor
 - Maximalizuje rekonstrukční chybu z náhodných vstupů
 - Generuje dekonvolucí obrázků z latentního prostoru
 - Diskriminátor
 - Minimalizuje chybu rozpoznávání skutečného a umělého obrázku
 - Trénování současně na skutečných a generovaných obrázcích
 - Jeho klasifikační chyba je výstupní chybou modelu
 - Problémy
 - Nestabilní trénování: jeden model se zlepší příliš rychle a druhý ho nemůže dohnat
 - Problém s počtem generovaných featur a s perspektivou - diskriminátor může hledat jen nějaké featury nezávisle na počtu
 - Mode collapse: generátor vytvoří distribuci s velkým rozptylem, diskriminátor dá důraz na jediný, nejpravděpodobnější bod
 - Těžko vygenerujeme konkrétní obrázek podle zadání, protože nemáme dobrou kontrolu nad vstupním šumem
- VAE (Variational AE)
 - Rozdělení latentního prostoru z generátoru na vektor středních hodnot a rozptylů
 - Tvorba rekonstrukční chyby ze dvou komponent
 - Klasická rekonstrukční chyba
 - Odchylka latentní distribuce od jednotkového normálního rozdělení

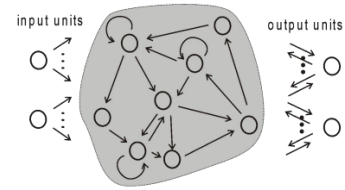
9 – Rekurentní neuronové sítě a jejich učení, neuroevoluce.

Rekurentní neuronové sítě

- Neuronové sítě s pamětí - v síti je alespoň jeden cyklus
 - Zachování dřívější informace
 - Větší důraz na nedávné vstupy
 - Vstupy (např. slova) se zakódují vždy na stejně dlouhý vektor
 - Sdílení parametrů mezi časovými kroky
 - Příznaky se opakovávají podobně jako v konvoluci obrázků
 - Např. věta se stejným významem, ale jiným slovosledem
 - Trénovací data: sekvence input-output párů
- Využití: NLP, překlad, speech recognition, image captioning
- BPTT (Backpropagation Through Time)
 - Trénování RNN, kdy se chyby napříč časovými kroky sčítají
 - Exploding/vanishing gradient problem
 - Rozbalení sítě, kumulace chyby, pak zpět složení a aktualizace vah
 - Truncated: chyba se kumuluje pouze pro podsekvenci
- Vanilla RNN (Elman net)
 - Feedforward síť s částečnou rekurencí
 - Vrstvy: vstup, skrytá vrstva, kontextová vrstva, výstup
 - Kontextová vrstva
 - Krátkodobě si pamatuje výstupy skryté vrstvy
 - Každý neuron má paměťovou buňku
 - Učení pomocí BPTT rozbalením
 - Predikce následujících slov ve větách (Elman 1990)
- Hopfield net
 - Content-addressable memory: ukládání dat
 - Všechny neurony mají jako vstup výstupy všech ostatních neuronů
 - Hebbovské nesupervizované učení
 - Posilování synapse mezi dvěma společně aktivovanými neurony (jako mozek)
 - Váha synapse je součet násobků vstupů obou neuronů
 - Provádí se v jednom kroku naráz, narozdíl od backpropagation
 - Konverguje (bez garance) k uloženému vzoru (energy landscape analogie)
 - Např. pro opravu poškozeného vstupu
 - Odolnost vůči výpadkům
 - Omezená kapacita (uložených vzorů asi $n/6$, kde n je počet neuronů)



- Echo state networks (dynamical reservoirs)
 - Black-box modelování složitých nelineárních systémů rekurencí
 - Signál vstupuje do RNN, šíří se a osciluje
 - Predikce časových řad (např. generátor zvuku)
 - Učení pouze vah výstupní vrstvy
 - Minimalizace MSE, backpropagation
 - Rekurentní část může mít přes 100 neuronů
 - Chybí dlouhodobá paměť
 - Naopak pro krátkodobé sekvence a vztahy je rychlejší Vanilla RNN
- LSTM (Long Short Term Memory)
 - Princip paměti
 - Short term memory: stav jednotlivé buňky složené z hradel
 - Long term memory: vnitřní stav sítě, určuje výstup
 - Do každého hradla vstupuje hodnota z paměti s určitou vahou
 - Typy hradel v buňce
 - Forget gate (1)
 - Aktuální vstup a short term určují, kolik informace z long term se zapomene
 - Vstup + short term, pak sigmoida
 - Input gate (2, 3)
 - Aktuální vstup a short term určují, kolik informace se přidá do long term
 - Vstup + short term, z nich sigmoida a tanh, které se pronásobí
 - Update gate (4)
 - Aktuální vstup a short term určují, kolik informace přejde do short term
 - Vstup + short term, pak sigmoida
 - Output gate (5)
 - Long term jde na aktuální výstup
 - Long term a výstup Update gate se pronásobí a určí následující hodnotu short term
 - V modelu může být použito několik buněk
 - Každá jednotka má poměrně dost vah (parametrů)
 - Učení backpropagací, všechno je diferencovatelné díky spojitosti
- GRU (Gated Recurrent Unit)
 - Nemá vnitřní stav buňky, k rekurenci používá long term memory
 - Jen dvě hradla
 - Update gate: kolik paměti se předá do dalšího kroku (sigmoida)
 - Reset gate: kolik paměti se zapomene (sigmoida)
 - Méně parametrů než LSTM
 - Lepší než LSTM na určitých malých datech
 - Omezenější schopnosti než LSTM
- Clockwork RNN
 - Skupiny Vanilla RNN modulů, každý s jinou frekvencí
- Stack RNN
 - Vanilla RNN, ale místo kontextuálních buněk zásobník či více paralelních



Neuroevoluce

- Využití evolučních algoritmů ke generování neuronových sítí
 - Kombinace evoluce spojitých vah a diskrétní topologie
 - Velmi obecná a flexibilní metoda učení
- Popis problému
 - Kódování: genom určující vlastnosti sítě, nějaká datová struktura
 - Fitness: kvalita jedince
 - Operátory
 - Křížení: musí produkovat korektní potomky (bez rekurence...)
 - Mutace
 - Selektce: ruleta, turnaj
- GNARL: dva typy mutace
 - Parametrická: váhy mutujeme Gaussovským šumem
 - Strukturální: přidávání a odebírání neuronů a spojů
- SANE (Symbiotic Adaptive Neuroevolution)
 - Koevoluce dvou samostatných populací
 - Populace neuronů s vahami
 - Fitness: kvalita 5 nejlepších sítí, kde se neuron použil
 - Populace blueprintů určujících spojení neuronů
 - Fitness: kvalita popisované sítě
- NEAT (Neuroevolution of Augmenting Topologies)
 - Komplexifikace: začíná se z malých topologií, které se postupně zvětšují
 - Genotyp
 - Neurony: pro každý neuron jeho typ (input, output, hidden)
 - Spoje: zdroj a cíl, váha, stav zapnutí, inovační číslo (kdy byl přidán)
 - Parametrická a strukturální mutace (bez mazání)
 - Gaussovské zašumění vah
 - Přidání spoje mezi dva nespojené neurony
 - Vypnutí spoje (aniž by se smazal)
 - Přidání neuronu navíc po cestě mezi dvěma již spojenými
 - Křížení podle inovačních čísel - zajišťuje kompatibilitu genomů a zesložštění
 - Niching
 - Rozdělení populace na druhy
 - Sítě podobných vlastností se kříží jen mezi sebou
 - Podobnost na základě inovačního čísla
 - Podle počtu stejných genů vč. inovačního čísla (?)
 - Dostatečně odlišný jedinec zakládá nový niche
 - Fitness sharing: umožní vývoj slabších jedinců a zabrání overfittingu
- HyperNEAT
 - U přímého kódování (každý spoj explicitně) je špatné škálování, proto HyperNEAT optimalizuje "DNA", ze které síť pak staví
 - Neurony umístěny pevně na substrát
 - n -rozměrná mřížka, každý neuron má své souřadnice
 - CPPN síť přiřazuje váhy spojům neuronů (to je ta DNA)
 - Funkce, která vytváří pravidelné vzory
 - Vstupem jsou souřadnice, výstupem hodnota v daném bodě
 - Složena hlavně z periodických a symetrických funkcí (Gauss, sinus)

- Umožňuje trénovat obří sítě - vzorkování z CPPN je snadné
- Substrát lze po natrénování zjemnit (zvětšit počet neuronů)

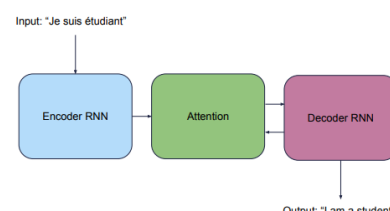
Tohle už do otázky nepatří, je to spíš k optimalizaci obecně (ověřeno u zkoušky 😊)

- Novelty search
 - Myšlenka neopakovatelnosti evoluce
 - Prostor prohledávání je plný lokálních minim
 - Při optimalizaci je třeba navštívit spoustu nečekaně důležitých míst
 - Proto nelze spoléhat jen na optimalizaci fitness
 - Curiosity driven learning: odměna za to, že se agent dostane na nové místo
 - Není třeba fitness, stačí hledat nové věci
- CMA-ES (Covariance Matrix Adaptation Evolution Strategy)
 - Strategie pro numerickou nelineární nekonvexní black-box optimalizaci
 - Kombinace evolučních a gradientních technik - překoná lokální minima
- ACO (Ant Colony Optimization)
 - Princip mravenců komunikujících pomocí feromonů
 - Agent nanáší cestou v prostoru feromon
 - Čím více feromonu je naneseno, tím častěji tudy budou agenti chodit
 - Problém lokálního minima - kombinace s tabu search
 - Např. hledání cest v grafech
- PSO (Particle Swarm Optimization)
 - Částice s rychlostním vektorem, udává směr pohybu
 - Každý jedinec má lokální nejlepší známé řešení, chce se k němu vrátit
 - Hejno má globální nejlepší řešení a všichni míří zároveň k němu
 - Balancování vlivu lokálního a globálního optima parametrem
 - Záleží na počáteční rychlosti, může dojít k oscilaci

10 – Transformery, pozornostní mechanismy, transfer a meta learning.

Pozornostní mechanismy

- Attention: napodobení kognitivní pozornosti na části vstupu a jejich vztahy
 - Zpracování textu: plnovýznamová slova, ne tolik gramatická
 - Zpracování obrazu: oblasti důležité pro popis, ne např. Pozadí
- Princip (self-attention)
 - Vstup
 - Několik hodnot X - příznaků obrazu nebo slov v textu
 - Aktuální hodnota x (jedno slovo), které chceme přiřadit attention ve vztahu k ostatním
 - Získání parametrů α : vektoru důležitostí vztahů v a ostatních hodnot
 - Key k : aktuální hodnota x převedená vahami
 - Query q : celý vstup X převedený vahami
 - Kombinací k a q získáme vektor α
 - Dot product attention
 - Skalární součin k a q
 - Následná nelineární transformace (tanh)
 - Normalizace softmax, aby vektor α byl jednotkový
 - Další mechanismy (aditivní, vážený, ...)
 - Výstup
 - Celý vstup X se převede vahami na value V
 - Lineární kombinace hodnot V s vektorem parametrů α
 - Výsledkem je attention hodnoty x ve vztahu ke zbytku vstupu
 - Učení vah převádějících i -tou hodnotu a celý vstup na k, q, v
- Typy attention mechanismů
 - Self-attention: keys a values jsou stejné, hledáme relevanci jednoho slova vůči ostatním (např. která slova odkazují na stejný koncept)
 - Global/soft attention: uvažuje celý vstup
 - Local/hard attention: uvažuje pouze část vstupu
 - Lepší soustředění např. na kus obrázku
 - Větší výpočetní složitost
- RNN Seq2seq
 - Aplikace attention na autoenkodér
 - Příklad překladu z angličtiny do francouzštiny
 - Attention blok dostává výsledek enkodéru a předchozí výstup dekodéru
 - Určuje důležitost vstupů do dekodéru
 - Pro každé slovo vstupu tak máme jeho důležitost vzhledem ke slovům výstupu
 - Lze vizualizovat jako matici, která ukazuje vztah anglických a francouzských slov - mimodiagonální vztahy ukazují, že model překládá lépe než jen "slovo od slova"



Transformery

- Zpracování sekvenčních dat, ale narozdíl od RNN paralelně
 - Rychlejší výpočet
 - Není vanishing gradient problem
 - Všechny attention bloky mají stejný query a key
- Využití pro překlad, sumarizaci textu, popis obrázků...
- Jednotlivá slova mají přidáný časový vektor, který kóduje jejich pozici ve vstupu
- Pozornostní mechanismy
 - Self-attention
 - Převážení embeddingů tak, aby jejich reprezentace zachycovala kontext
 - Užitečné pro mnohoznačná slova
 - Nevyžaduje učení, lze případně trénovat parametrizací k , q , v matic
 - Multi-head attention
 - Slovo může mít vztah s jinými v různých kontextech (významy, gramatika)
 - Každý vstup má několik attention modulů
 - Jejich výstupy se spojí a v transformeru pokračují do dense vrstev
 - Masked attention
 - Zakrytí vstupů následujících po aktuálním slovu v dekodéru
 - Model tak nemůže podvádět pohledem do budoucnosti
 - Cross attention
 - V dekodéru bereme key a values ze stacked enkodérů
 - Queries jsou z předchozího výstupu stacked dekodéru
- Transformer
 - Stacked enkodér a dekodér s multi-head attention
 - Dekodér s masked attention a cross attention
 - Mnoho spojení pro potlačení vanishing gradient problému
 - Na výstupu lineární dekodovací vrstva: určuje pravděpodobnost slov ze slovníku
 - Vysoké paměťové nároky
 - Kvadratická složitost attention vrstvy vzhledem ke vstupu
 - Špatná rozšiřitelnost na dlouhé sekvence
 - Složitá implementace
- BERT
 - Trénink
 - Masked language model: doplňování zamaskovaných slov
 - Next sentence prediction: jestli věta následuje v kontextu jinou
 - Seq2seq model s transformer architekturou (stacked enc/dec)
 - Jazykový základ s možností fine-tuningu
- Performer
 - Řeší problém škálovatelnosti attention bloků
 - Aproximuje key a query matice (před násobením s value je tam nelineární softmax, ten je třeba simulovat)
 - Lineární vzhledem k délce sekvence

- Synthesizer
 - Key a query se dodávají trénovatelnou sítí - umělý vstup
 - Random synthesizer: náhodná matice
- Reformer
 - Lokální sémantické hashování - sousedé z podobných kontextů mají podobný hash
 - Podobnosti jsou tedy předpočítané
 - Rychlejší, ale náročnější na paměť

Meta learning

- Učení na dvou úrovních
 - Učení na konkrétním úkolu (rychlé)
 - Úprava učicí strategie na základě zkušenosti z více úkolů (dlouhodobé)
- Příklady
 - Neuroevoluce
 - Meta level: evoluční algoritmus
 - Bottom level: tvořená neuronová síť
 - Hyper networks
 - Meta level: evoluční algoritmus + malá síť
 - Bottom level: velká síť
 - Learning to learn by gradient descent by gradient descent
 - Meta level: LSTM k zapamatování sekvence updatů vah
 - Bottom level: učená síť
- Optimalizace hyperparametrů
 - Grid search: systematické prohledávání parameter gridu
 - Random search: náhodné prohledávání param. grid, docela dobré
 - Bayesovská optimalizace: informované prohledávání na základě kvality hyperparametrů pro minimalizaci výstupní chyby
 - HyperBand
 - Bandit-based přístup
 - Kombinuje alokaci zdrojů a brzké zastavení
 - Každé konfiguraci přiřadí budget několika iterací
 - Periodicky zahazuje horší polovinu konfigurací, když dojde budget
 - Budget pak přerozděluje mezi přeživší, dokud nezůstane jeden
- Metadatabáze
 - Uchovává metadata k trénování různých problémů
 - Jaké metody mají jaké výsledky na jaké problémy
 - Lze doporučovat vhodné algoritmy, ale kompatibilita není zajištěna
- MAML (Model Agnostic Meta Learning)
 - Přístup pro nastavování parametrů nezávisle na modelu
 - Update parametrů gradientním sestupem na základě vyhodnocení problému ze sady
- NAS (Neural Architecture Search)
 - Automatické učení a evoluce topologie neuronových sítí
 - Stavový prostor, kde přechody reprezentují např. fully connected či konvoluční vrstvy
 - Velký prostor a náročné ohodnocení fitness, patří sem neuroevoluce

- AutoML
 - Kromě selekce modelu a tréninku i předzpracování dat
 - Kritéria: kvalita, rychlost, jednoduchost, vysvětlitelnost
- Few-shot learning
 - Učení na velmi malých datasetech
 - Support set: trénovací data, na kterých se model naučí učit
 - Využití znalosti podobných známých případů
 - Využití znalostí o učení - nedostatkem dat nutíme model generalizovat
 - Využití znalostí o datech - distribuce, variabilita

11 – Ensemble metody: rozdíl mezi základními metodami (např. Bagging, Boosting).

Ensemble metody

- Použití více modelů stejného typu k dosažení lepších výsledků
- Kvalitnější predikce, zmírnění overfittingu
- Bagging snižuje rozptyl
- Boosting snižuje bias
- Rozhodovací stromy
 - Klasifikace i predikce
 - Konstrukce pomocí ID3: rekursivní výběr nejlepšího rozdělení vzorků podle entropie či Gini indexu
 - Jednoduché s rychlým učením a dobrou interpretovatelností
 - Špatná robustnost - malá změna dat může ústít v úplně jiný strom
 - Náchylné na overfitting

Bagging (bootstrap aggregating)

- Princip
 - Trénink několika nezávislých modelů
 - Kombinace predikcí průměrováním či hlasováním
 - Každý model se tvoří na bootstrapped vzorku (výběr s opakováním)
 - Výhody
 - Menší rozptyl výstupu díky průměrování
 - Méně overfittingu díky tréninku na různých podmnožinách dat
 - Nevýhody
- Random forest
 - Konstruuje mnoho mělkých stromů (weak learners)
 - V každém uzlu se dělí pouze podle náhodné podmnožiny příznaků
 - Hyperparametry: počet stromů, velikost bootstrapu, hloubka stromů

Boosting

- Princip
 - Trénink weak learnerů postupně, každý se snaží napravit předchůdce
 - Použití mělkých stromů i jen decision stumps, naivního bayese, kNN
- AdaBoost (Adaptive Boosting)
 - Vážení vzorků a jednotlivých podmodelů
 - Na začátku mají všechny vzorky stejnou váhu
 - Podle nich vážená evaluace weak learnera
 - Zvýšení vah špatně klasifikovaných, aby se na ně další model zaměřil
 - Toto je ta adaptivita
 - Predikce je váženým hlasem či sumou dle kvality jednotlivých modelů
 - Výhody
 - Všestrannost i na komplexní problémy

- Nevýhody
 - Riziko přeučení, hlavně na zašuměných datech (řešení regularizací)
 - Složitost výběru klasifikátoru a hyperparametrů
- XGBoost
 - Spojitá vysvětlovaná proměnná (pro klasifikaci je pravděpodobnost příslušnosti vzorku ke třídě)
 - Používá rozhodovací stromy, počet daný hyperparametrem
 - Gradient boosting
 - Navazující modely predikují nevysvětlené reziduum z minulého, ne přímo vysvětlovanou proměnnou
 - Snažíme se tedy dostat ne přímo výstup, ale co nejmenší chybu
 - Modely jsou následně složeny a výsledek je jejich součtem
 - Aproximace ztrátové funkce pomocí Taylorova polynomu 2. Stupně
 - Umožňuje použít loss bez 2. derivace
 - Navíc regularizace daná složitostí stromu (L1, L2)

12 – Jádrové metody: jádrová regrese, báзовé funkce, Support Vector Machine (SVM): separabilní a neseperabilní případ.

Základní metody

- Lineární regrese
 - $Y = w^T X$: matice dat X , vysvětlovaný vektor Y , váhy w , náhodná proměnná ε
- OLS (Ordinary Least Squares)
 - Minimalizace reziduálního součtu čtverců $RSS(w) = \|Y - Xw\|^2$
 - Hledání minima pomocí $\nabla RSS(w) = 0$
 - Minimum z normální rovnice $X^T Y - X^T X w = 0$
 - Pro regulární $X^T X$ je řešení $w_{OLS} = (X^T X)^{-1} X^T Y$
- Hřebenová regrese
 - Regularizovaný $RSS_\lambda(w) = \|Y - Xw\|^2 + \lambda \sum_{i=1}^p w_i^2$
 - Sumu vyjádříme pomocí jednotkové matice s první 0 jako $\lambda w^T I' w$
 - Nově řešení $w_\lambda = (X^T X + \lambda I')^{-1} X^T Y$

Báзовé funkce

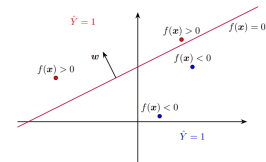
- Lineární báзовá expanze
 - Nahrazení původních příznaků za transformované
 - Lze pak použít nelineární transformace, model je ale stále lineární
 - Transformace
 - Báзовé funkce $\phi_1 \dots \phi_M$: lineárně nezávislé, zobrazují vektor příznaků do \mathbf{R}
 - Vektor báзовých funkcí ϕ : z p -dimenzionálních původních příznaků do M -dimenzionálních transformovaných příznaků
 - Matice zobrazených příznaků $\Phi = (\phi(x_1)^T \dots \phi(x_N)^T)$
 - Model $Y = \Phi w + \varepsilon$ zůstává lineární
 - Minimalizace s regularizací: $RSS_\lambda(w) = \|Y - \Phi w\|^2 + \lambda w^T w$
 - Báзовých funkcí bývá víc než příznaků, proto je dobrá regularizace
 - Normální rovnice $\Phi^T Y - \Phi^T \Phi w - \lambda w = 0$
 - Řešení $w_\lambda = (\Phi^T \Phi + \lambda I)^{-1} \Phi^T Y$
 - Predikce $Y' = w_\lambda^T \phi(x)$
- Běžné jádrové funkce
 - Identita: původní lineární model
 - Kvadrát, součiny příznaků: polynomiální regrese
 - Další nelineární funkce - odmocnina, logaritmus, sinus...
 - Indikátor hodnoty příznaku v množině/intervalu: rozdělení prostoru na nezávisle modelované části

Kernel trick

- Duální reprezentace
 - Váhy vyjádříme pomocí vektoru $\alpha \in \mathbf{R}^N$ jako $w = \Phi^T \alpha$
 - Minimalizujeme pak $RSS_\lambda(\alpha) = \|Y - \Phi \Phi^T \alpha\|^2 + \lambda \alpha^T \Phi \Phi^T \alpha$
 - Tato minimalizace je ekvivalentní minimalizaci $RSS_\lambda(w)$
- Gramova matice $G = \Phi \Phi^T \in \mathbf{R}^{N, N}$
 - Symetrická matice $N \times N$
 - Dosadíme do minimalizace $RSS_\lambda(\alpha) = \|Y - G\alpha\|^2 + \lambda \alpha^T G \alpha$
- Jádrová funkce $k(x, y) = \phi(x)^T \phi(y)$ pro všechna $x, y \in \mathbf{R}^p$
 - Složky Gramovy matice jsou definovány jádrovou funkcí
 - $G_{i,j} = (\Phi \Phi^T)_{i,j} = \sum_{t=1}^M \phi_t(x_i) \phi_t(x_j) = k(x_i, x_j)$
- Odhad α
 - Minimalizace $RSS_\lambda(\alpha)$
 - Normální rovnice $G(Y - G\alpha - \lambda\alpha) = 0$
 - Gramova matice je pozitivně semidefinitní, proto $(G + \lambda I)$ je poz. Definitní
 - Odhad $\alpha = (G + \lambda I)^{-1} Y$
 - Vyjádřený jádrovou funkcí, protože tam máme G
 - Predikce $Y = \sum_{i=1}^N \alpha_i k(x_i, x) = \alpha^T k(x)$
- Vlastnosti reprezentace
 - Díky jádrové funkci nemusíme definovat báze funkce
 - Můžeme vyjádřit transformace do prostorů nekonečné dimenzionality
 - Odhad pomocí jádrového triku má vzhledem k maticové inverzi složitost pouze $O(N^3)$ namísto odhadu přes báze funkce, který je $O(N^2)$
- Běžné jádrové funkce - musí být pozitivně semidefinitní
 - Kvadratické jádro $k(x, y) = (x^T y + 1)^2$ odpovídá šesti bázevým funkcím
 - Gaussovské jádro
 - Jádro kosinové podobnosti: pro porovnávání dokumentův bag of words
 - Složené z TF-IDF (term frequency - inverse document frequency) funkce
- Využití v kernel machines
 - Můžeme použít logistickou regresi s lineární bázeovou expanzí
 - Decision boundary je pak nelineární a oddělí složitější data

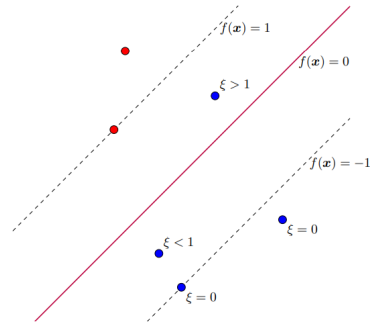
SVM

- Lineárně separabilní případ
 - Discriminant function $f(x) = w^T \phi(x) + w_0$
 - Funkci lze vyjádřit jádrovým trikem
 - Vektor w definuje normálu přímky rozhodovací hranice
 - Body jsou klasifikovány do dvou tříd
 - $f(x) > 0$: třída 1
 - $f(x) < 0$: třída -1
 - Large margin principle
 - Najdeme takovou rozhodovací hranici, pro kterou jsou její vzdálenosti od nejbližšího bodu maximální
 - Vektor w přeškálujeme koeficientem tak, aby jeho hledaná velikost byla vždy alespoň 1



- Hledáme pak jednoduše jeho minimalizaci za podmínky, že všechny body jsou správně klasifikovány se vzdáleností od hranice alespoň 1
- Lineárně neseeparabilní případ
 - Původní řešení neexistuje
 - Přidáme penalizaci, která se zvyšuje tím, čím dál na špatné straně hranice se každý bod nachází
 - Slack variables $\xi_i \geq 0$ pro každý bod
 - $\xi_i = 0$, pokud je bod za správnou stranou marginu
 - $\xi_i < 1$ pro bod na správné straně hranice, ale uvnitř marginu
 - $\xi_i > 1$ pro bod na špatné straně hranice
 - Podmínku správné klasifikace zlehčíme pomocí slack variables $Y_i f(x_i) \geq 1 - \xi_i$ pro všechny body $i \in \mathbb{R}^N$
 - Minimalizace pak odpovídá

$$\min_{w, w_0, \xi} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^N \xi_i$$
 s regularizačním parametrem C : kolik chybných klasifikací tolerujeme
- Support vector machine



$$f(x) = \sum_{j=1}^N a_j Y_j k(x, x_j) + w_0$$

- Což je sparse vector machine, kde $\alpha_j = a_j Y_j$
 - Vector machine: středy jsou dány přímo trénovacími body
 - Sparse: $a_j = 0$ pro mnoho bodů
- Support vektory: $a_j > 0$
 - Body nejbližší rozhodovací hranici, nejtěžší klasifikace
 - Mají přímý vliv na polohu hranice
 - Čím větší je reg. parametr C , tím řidší je řešení
- Predikce odpovídá znaménku $f(x)$
- Proměnné a_i najdeme jako Lagrangeovský duální problém maximalizace

$$\tilde{L}(a) = \sum_{i=1}^N a_i - \frac{1}{2} \sum_{i,j=1}^N a_i a_j Y_i Y_j k(x_i, x_j)$$

s podmínkami

$$0 \leq a_i \leq C \quad \sum_{i=1}^N a_i Y_i = 0$$

- Souvislost s KKT (prostě minimalizace funkce více proměnných s vazbami)
 - 0. Body jsou správně klasifikovány (se zlehčením slackness)
 - 1. Gradient Lagrangiánu je roven 0 (podm. 1. derivace)
 - 2. Pro každý multiplikátor a_j (aktivní a neaktivní vazby)
 - Buď $a_j = 0$
 - Nebo x_j leží na marginu či uvnitř: pak je to support vector
 - 3. Lagrangeovy multiplikátory $a_j \geq 0$: správný směr bodu od hranice

13 – Algoritmy pro doporučování: základní přístupy a způsob vyhodnocení kvality, faktorizační metody pro doporučování.

Základní přístupy

- Aktéři
 - Uživatel: např. návštěvník e-shopu
 - Položka: např. produkt na e-shopu
- Princip doporučovacích algoritmů
 - Uvažujeme historické interakce mezi uživateli a položkami
 - Hodnotíme aktéry na základě jejich podobnosti
 - Doporučení jsou personalizovaná
- Módy doporučování
 - Položky doporučované uživatelům (nejběžnější)
 - Uživatelé položkám
 - Položky položkám (co si přikoupit na e-shopu k tomu)
 - Uživatelé uživatelům (sociální sítě)
- Interakce aktérů
 - Explicitní: např. ohodnocení filmu hvězdičkami
 - Implicitní: např. vložení produktu do košíku
- Přístupy
 - Collaborative filtering: pokud se uživatelé shodnou na jedné věci, křížově jim doporučíme zájmy druhého
 - Content-based filtering: doporučujeme položky podobné těm, se kterými uživatel interagoval v minulosti
 - Hybridní metody
 - Demografické metody: podle lokality, věku, ...
 - Užití metody: tvorba užití funkce pro každého uživatele
- kNN
 - Collaborative: hledá k nejpodobnějších uživatelů
 - Content-based: hledá k nejpodobnějších položek těm kladně hodnoceným
 - Volba k výrazně ovlivňuje kvalitu a rychlost doporučování

Způsob vyhodnocování kvality

- Cumulative gain
$$CG_k = \sum_{i=1}^k rel_i$$
 - Hodnotí relevanci prvních k položek
 - Neřeší jejich řazení

- Discounted cumulative gain

$$DCG_k = \sum_{i=1}^k \frac{rel_i}{\log_2(i+1)}$$

- Relevance logaritmičsky klesá vzhledem k pozici položky
- Lze ještě posílit vliv prvních položek použitím exponenciální funkce
- Precision@N
 - TP / (TP + FP)
 - Kolik z vrácených položek je skutečně relevantních
 - Počítá se jen z prvních N položek
- Recall@N
 - TP / P
 - Kolik ze skutečně relevantních položek bylo vráceno
- F₁ skóre
 - Harmonický průměr precision a recall (2 * prec * rec) / (prec + rec)
- AUC (Area Under Curve)
 - Plocha pod ROC: TPR a FPR podle měnící se hranice rozhodování

Faktorizační metody

- Matice interakcí: explicitní či implicitní síla pozitivní interakce uživatele a položky
- Princip faktorizace

$$R = UV^T$$
 - Rozložení na odvozené matice latentních atributů uživatelů a položek
 - Např. preference žánru či režiséra
 - Predikce jako skalární součin latentního vektoru uživatele a položky
 - Získáme tak predikci i pro položku bez interakce - řeší řídkost pův. matice
- Optimalizace faktorizace
 - Minimalizujeme rozdíl mezi původní a rekonstruovanou maticí podle MSE

$$\operatorname{argmin}_{U,V} \sum (r_{ij} - u_i^T v_j)^2$$
 - Navíc regularizace kvadrátu normy vektorů s koeficientem λ

$$+ \lambda (\sum_x |u_x|^2 + \sum_y |v_y|^2)$$
- ALS (Alternating Least Squares)
 - Inicializace U a V náhodně
 - Střídavě zafixujeme jednu z U a V, druhou učíme a minimalizujeme MSE
- Modelování implicitní zpětné vazby
 - Chceme aby velké číslo znamenalo interakci a 0 neznalost, ne nezáměr
 - Binární interakční matice P
 - Confidence matrix C s jistotou interakce
 - Minimalizace je pak

$$\min_{U,V} \sum_{i,j} C_{ij} (P_{ij} - u_i^T v_j)^2 + \lambda |u_i|^2 + \lambda |v_j|^2$$
- Omezení a problémy ALS
 - Cold start: nový uživatel či položka nemají data o historické interakci
 - Při extrémně řídké matici interakcí není doporučení kvalitní
 - Náročnost maticové inverze



14 – Principy bayesovského modelování - pojmy model, apriorní a posteriorní distribuce.

Exponenciální třída distribucí, konjugovaná apriorní a jejich význam v bayesovském odhadu. Příklad konjugovaného apriorního.

Principy bayesovského modelování

- Distribuce
 - Hustota pravděpodobnosti/Pravděpodobnostní funkce
 - Distribuční funkce
 - Střední hodnota
 - Rozptyl
 - Vícerozměrné
 - Sdružená hustota
 - Marginální hustota $f(x) = \int f(x, y) dy$
 - Podmíněná hustota $f(x|y) = f(x, y) / f(y)$
 - Řetězové pravidlo $f(x, y) = f(x|y) f(y) = f(y|x) f(x)$
- Bayesova věta
$$\pi(\theta|x) = \frac{f(x|\theta)\pi(\theta)}{f(x)}, \quad f(x) > 0$$

pomocí proporcionality jen

$$\pi(\theta|x) \propto f(x|\theta)\pi(\theta)$$
- Pojmy modelování
 - Apriorní distribuce $\pi(\theta)$
 - Počáteční znalost o studované proměnné před zohledněním dat
 - Inicializace ovlivní ochotu ke změně (Diracovu deltu nezměníme)
 - Model $f(x|\theta)$
 - Popis naměřených dat (likelihood, věrohodnost)
 - θ je parametr distribuce (třeba p Bernoulliho rozdělení)
 - Posteriorní distribuce $\pi(\theta|x)$
 - Upravená znalost proměnné po započtení dat
- Sekvenční odhad
 - i.i.d. měřené hodnoty y_t determinované vektorem x_t a konstantním parametrem θ , které chceme oba odhadnout
 - Stačí upravit model na základě nového pozorování
 - Nemusíme přepočítávat vše na základě rozšířených dat
 - Nebo se omezovat na sliding window dat, abychom to stíhali
 - Provedeme to aplikací bayesovy věty na apriorní distribuci
 - $\pi(\theta|y_{0:t}, x_{0:t}) \propto f(y_t|x_t, \theta) \pi(\theta|x_{0:t-1}, y_{0:t-1})$
 - Modely za delší čas lze nejprve pronásobit a až pak updatovat apriorní

- Zajímá nás bodový odhad parametru θ modelu
 - Střední hodnota $E\theta$
 - Modus (MAP), medián
- A jistota správnosti (věrohodnost) odhadu: rozptyl $\text{var}\theta$

Exponenciální třída distribucí

- Při aplikaci Bayesovy věty je problém odvodit aposteriorní distribuci
 - Při sekvenčním odhadu se může stále zesložitovat
- Model proto volíme tak, aby aposteriorní distribuce byla
- Mějme náhodnou veličinu y podmíněnou veličinou x a parametrem θ
- Exponenciální třída distribucí: distribuce s hustotou ve tvaru

$$f(y|x, \theta) = h(y, x) g(\theta) \exp[\eta^T T(y, x)]$$
 - $\eta = \eta(\theta)$: parametr
 - $T(y, x)$: suficientní statistika fixního rozměru (funkce náhodných veličin)
 - $h(y, x)$: známá funkce
 - $g(\theta)$: normalizační funkce (aby se to integrovalo na 1)
- Kanonická: $\eta = \eta(\theta) = \theta$

Konjugovaná apriorní

- Konjugovaná apriorní distribuce θ k: hustota ve tvaru

$$\pi(\theta) = q(\xi, \nu) g(\theta)^\nu \exp[\eta^T \xi]$$
 - Konjugovaná k distribuci $(y|x, \theta)$ s rozdělením z exp. třídy
 - ξ : parametr s rozměrem jako $T(y, x)$
 - ν : kladný reálný parametr
 - $q(\xi, \nu)$: známá funkce
 - $g(\theta)^\nu$: normalizační funkce
- Běžné konjugované distribuce

Model	Příklad použití	Konjugované apriorno
Normální se známým rozptylem	Všude možné :-)	Normální
Normální s neznámým rozptylem	Všude možné :-)	Normální inverzní-gama
Bernoulliho	Úspěch-neúspěch (mince, spolehlivost)	Beta
Binomický	Úspěch-neúspěch (mince, spolehlivost)	Beta
Poissonův	Řídké jevy (telefony, částice ve fyzice, doprava)	Gama
Multinomický	Klasifikace do více tříd (kostka, spolehlivost)	Dirichletovo

- Beta distribuce: pravděpodobnost úspěchu v binomickém rozdělení, používá gamma funkci (rozšíření faktoriálu na komplexní obor)
- Použití v bayesovském odhadu
 - Bayesova věta se zjednoduší na přičtení suficientní statistiky a inkrementaci hyperparametrů

$$\xi_t = \xi_{t-1} + T(y_t, x_t)$$

$$\nu_t = \nu_{t-1} + 1$$

15 – Stavové modely: rovnice pro vývoj stavu a rovnice měření, rozdíly mezi nimi. Bayesovský sekvenční odhad stavových modelů a jejich vliv na apriorní distribuci (znalost). Možnosti odhadu stavů v případě nelinearity (pouze vyjmenovat).

Stavové modely

- Stavový model
 - Tři veličiny
 - Nepozorovatelný stav x_t : odhadujeme
 - Vstup/řídící veličina u_t : známe
 - Výstup z_t : pozorujeme, je determinovaný stavem a řídící veličinou
 - Obecný zápis
 - Rovnice stavu: $x_t = f_t(x_{t-1}, u_t)$
 - Rovnice měření: $z_t = g_t(x_t)$
 - Lineární stavový model
 - Rovnice stavu: $x_t = A_t x_{t-1} + B_t u_t + v_t$
 - Rovnice měření: $z_t = H_t x_t + w_t$
 - Kde v_t, w_t jsou šумы stavu a měření
 - Pro časově invariantní model platí $A_t = A$, totéž pro B, H
- HMM (Hidden Markov Model)
 - Markovský proces: model splňující markovskou podmínku bezpaměťovosti
 - Skrytý markovský proces: není přímo pozorovatelný, lze na něj nahlížet jinou veličinou
 - Pro spojitý proces

Kalmanův filtr

- Časově invariantní stavový model
 - Šumové proměnné i.i.d. centrované v 0
 - $v_t \sim N(0, Q)$
 - $w_t \sim N(0, R)$
 - Odtud máme rozdělení stavu a měření
 - $x_t \sim N(Ax_{t-1} + Bu_t, Q)$
 - $z_t \sim N(Hx_t, R)$
- Model z_t je normální - konjugované apriorno je také normální
 - $\pi(x_t | z_{0:t-1}, u_{0:t-1}) = N(x_{t-1}^+, P_{t-1}^+)$
 - Kovariance P_{t-1}^+ dává míru neurčitosti odhadu stavu modelu

- Dva kroky filtrování
 - Predikce: získání odhadu nového stavu
 - Využití předchozího aposteriora (nového apriorního)

$$\pi(x_t | z_{0:t-1}, u_{0:t}) = \int p(x_t | x_{t-1}, u_t) \pi(x_{t-1} | z_{0:t-1}, u_{0:t-1}) dx_{t-1}$$
 - Násobíme normální distribuce modelu a apriorního, marginalizujeme
 - Předpověď následujícího stavu pomocí stavové rovnice

$$x_t^- = Ax_{t-1}^+ + Bu_t,$$

$$P_t^- = AP_{t-1}^+ A^T + Q$$
 - Díky konjugovanosti stačí upravit hyperparametry
 - Ty odpovídají stavu a kovarianci
 - Kovariance se zvětší (lin. transformace náhodné veličiny)
 - $\text{var}(AX + B) = A \text{var}(X) A^T$
 - Update: oprava odhadu podle pozorování
 - Pomocí Bayesovy věty

$$\pi(x_t | z_{0:t}, u_{0:t}) \propto f(z_t | x_t) \pi(x_t | z_{0:t-1}, u_{0:t})$$
 - Model zapíšeme jako distribuci z exponenciální třídy, najdeme apriorní
 - Bayesovský update je pak součtem hyperparametrů a suficientní statistiky
 - V updatu kovariance hraje roli Kalmanovo zesílení
 - Optimální Kalmanovo zesílení minimalizuje MSE
 - Čím větší zesílení, tím citlivější filtr - víc reaguje na data, ale méně proto filtruje šum
 - Update lze odvodit i nebayesovsky - původně tak filtr vznikl

Odhad stavů v případě nelinearity

- Stavový model v obvyklém tvaru, ale alespoň jedna funkce není lineární

$$x_t = f_t(x_{t-1}, u_t)$$

$$z_t = g_t(x_t)$$
- Slabá nelinearita: lokální linearizace derivacemi v EKF (Extended KF)
- Silná nelinearita: UKF (Unscented KF) s aproximací Taylorem
- Particle filter: Monte Carlo vzorkování
 - Generujeme velký počet náhodných vektorů x_t
 - Těm, které jsou nejpravděpodobnější pro z_t se zvýší váhy
 - Hledá se v jejich okolí

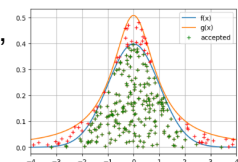
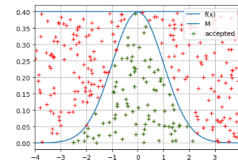
16 – Rejection sampling (RS) a importance sampling (IS): důvody používání RS a IS, jejich základní principy a rozdíly, efektivita práce se vzorky. Stanovení vah v IS a možnosti jejich normování.

Základní principy

- Z definice integrálu přístup pomocí sčítání malých obdélníků
 - V definici máme ekvidistantní rozdělení intervalu
- Monte Carlo integrace
 - Body (hranice obdélníků) vzorkuje z rovnoměrného rozdělení
 - Zákon velkých čísel zaručuje konvergenci ke skutečné hodnotě
- Využití vzorkování pro úpravu apriorní, když není konjugované k modelu

Rejection sampling

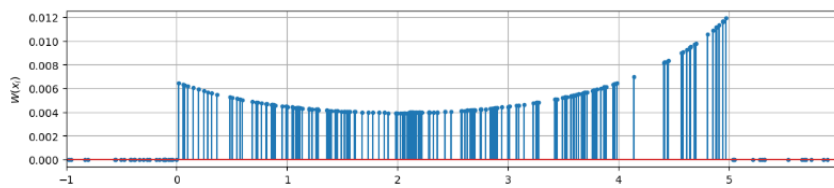
- Proposal distribuce: lze z ní snadno vzorkovat softwarově
- Primitivní Monte Carlo
 - Vzorkujeme bod x' z rovnoměrného rozdělení $\text{Unif}(a, b)$
 - Vzorkujeme bod u z rovnoměrného rozdělení $\text{Unif}(0, m)$
 - Přijmeme vzorek, pokud $0 < u < f(x')$
- Rejection sampling
 - Chceme proposal přiblížit skutečné distribuci, abychom nezahazovali zbytečně tolik vzorků
 - Proposal g se bude blížit skutečné distribuci
 - Zvolíme vhodné M tak, aby $Mg(x') \geq f(x')$
 - Přeshkálovaná hustota g tak pokryje celou odhadovanou hustotu f
 - Algoritmus
 - Vzorkujeme bod x' z proposal distribuce g
 - Vzorkujeme bod u z rovnoměrného rozdělení $\text{Unif}(0, 1)$
 - Přijmeme vzorek, pokud $u \leq f(x') / Mg(x')$
 - Díky tomu můžeme navzorkovat všechna u ze stejného rovnoměrného rozdělení najednou a přeshkálovat až při výběru
 - Pravděpodobnost přijetí u RS je $1/M$
 - Efektivnější než MC, ale když je hustota velmi koncentrovaná nebo nabývá nízkých hodnot, je pravděpodobnost přijetí malá a není to efektivní



Importance sampling

- Rejection sampling zahazuje vzorky, což znamená spoustu zbytečné práce

- Importance sampling místo zahození přiřadí vzorku váhu



- Pokud je $g(x) > 0$ všude tam, kde $f(x) > 0$, pak platí

$$\int f(x) dx = \int g(x) \frac{f(x)}{g(x)} dx = \int g(x) w(x) dx$$

$$w(x) = \frac{f(x)}{g(x)}$$

- Každému vzorku tak přiřadíme váhu $w(x) = f(x) / g(x)$
- Odhad střední hodnoty získáme jako vážený průměr vzorků
- Normované váhy
 - Získané váhy se nesčítají na 1
 - Lze je znormovat a střední hodnotu odhadnout opět váženým průměrem
 - Odhad je vychýlený, ale může mít nižší varianci
 - Navíc nezávisí na normalizační konstantě
- SIS (Sequential Importance Sampling)
 - Sekvenční odhadování stavových modelů
 - Apriorní distribuci navzorkujeme jako empirickou distribuci
 - Predikce: přenásobíme vzorky modelem
 - Update: přepočítáme váhy a znormalizujeme je
 - Váhy ale časem zdegenerují na několik málo vzorků
 - SIR (Sequential Importance Resampling) při predikci provádí nové vzorkování z apriorní distribuce

17 – QR rozklad: metody výpočtu, použití při výpočtu odhadu metodou nejmenších čtverců, QR algoritmus pro hledání vlastních čísel.

QR rozklad

- Ortogonální vektor: $x^T y = 0$
- Ortogonální matice: $Q^T Q = I$
 - Její sloupce jsou vzájemně ortogonální a mají jednotkovou velikost
 - Transpozice je inverzí
 - Transpozice je také ortogonální
 - Součin ortogonálních matic je ortogonální matice
 - Násobení vektoru OG maticí nemění jeho normu
 - Vhodné pro strojovou aritmetiku (není katastrofické krácení)

- QR rozklad

$$A = QR = \begin{pmatrix} Q_L & Q_R \end{pmatrix} \begin{pmatrix} R_S \\ \Theta \end{pmatrix}$$

- $A \in \mathbb{R}^{m,n}$
- $Q \in \mathbb{R}^{m,m}$: ortogonální matice
- $R \in \mathbb{R}^{m,n}$: horní trojúhelníková matice
- Výpočet QR rozkladu
 - Pro QR rozklad $A = QR$ platí
$$Q^T A = R$$
 - Chceme najít Q^T , která vynuluje prvky A pod diagonálou a získáme tak R
 - Lze získat posloupnost ortogonálních matic $Q_1 \dots Q_k$ matic, které vynulují prvky pod diagonálou A postupně
 - Q pak získáme z $Q^T = Q_k \dots Q_2 Q_1$ (součin OG je OG)

Givensovy rotace

- Hledáme ortogonální matici, která vektor otočí tak, aby se jeden jeho prvek vynuloval

$$\begin{pmatrix} \alpha & \beta \\ -\beta & \alpha \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \pm \|x\| \\ 0 \end{pmatrix} = \begin{pmatrix} \pm \sqrt{a^2 + b^2} \\ 0 \end{pmatrix}$$

- První prvek odpovídá normě, protože vektor se pouze otočil a jeho norma se nezměnila
- Získáme podmínku na α, β
$$-a\beta + b\alpha = 0 \Leftrightarrow a\beta = b\alpha$$
- Umocníme na druhou a přidáme podmínku na normu sloupce v OG matici rovnou 1
$$a^2 \beta^2 = b^2 \alpha^2,$$
$$\alpha^2 + \beta^2 = 1.$$
- Odtud odvodíme α, β = funkce a, b

- Lze také jako $\alpha = \cos\varphi$, $\beta = \sin\varphi$
- Rozšíříme na nulování libovolného prvku matice

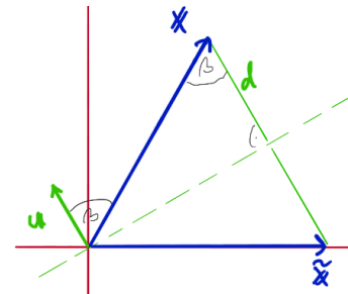
$$\underbrace{\begin{pmatrix} \alpha_2 & 0 & -\beta_2 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \beta_2 & 0 & \alpha_2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}}_{Q_2} \underbrace{\begin{pmatrix} a_2 & \bullet & \bullet \\ 0 & \bullet & \bullet \\ b_2 & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \end{pmatrix}}_{Q_1 A} = \underbrace{\begin{pmatrix} a_3 & \bullet & \bullet \\ 0 & \bullet & \bullet \\ 0 & \bullet & \bullet \\ b_3 & \bullet & \bullet \\ \bullet & \bullet & \bullet \end{pmatrix}}_{Q_2 Q_1 A}$$

- Využití pro QR rozklad je stejně náročné jako GEM (princip je stejný), ale je numericky stabilní díky zachování normy rotovaných vektorů

Householderovy reflexe

- Zrcadlíme vektory podle nadroviny procházející počátkem, aby ležely na ose x
- Vynulujeme tím najednou všechny prvky kromě prvního
- Hledáme ortogonální matici P

$$Px = P \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix} = \begin{pmatrix} \pm \|x\| \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$



- Zrcadlíci nadrovinu popisuje normálový vektor u
 - $|u| = 1$
- Vzdálenost x od nadroviny je d
- Zrcadlový obraz posuneme o $2d$ zpět

$$\tilde{x} = x - 2du.$$

- Odvodíme vzdálenost d

$$d = u^T x$$

- $\cos\beta$ jako skalární součin u, x
- $\cos\beta$ jako trigonometrie d a $|x|$
- Odtud matice P

$$\tilde{x} = x - 2u^T x u = x - 2uu^T x = \underbrace{(I - 2uu^T)}_P x$$

- Nakonec odvodíme složky u

$$x - 2uu^T x = \alpha e_1 \quad \Leftrightarrow \quad u(2u^T x) = x - \alpha e_1$$

$2u^T x$ je číslo

a odtud s normalizací na jednotkový u

$$u = \pm \frac{x - \alpha e}{\|x - \alpha e\|} = \pm \begin{pmatrix} \frac{x_1 - \alpha}{\|x - \alpha e\|} & \frac{x_2}{\|x - \alpha e\|} & \dots & \frac{x_m}{\|x - \alpha e\|} \end{pmatrix}^T$$

- Pro QR rozklad konstruujeme posloupnost $P_1 \dots P_n$

$$P_1 A = P_1 \begin{pmatrix} \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet \end{pmatrix} = \underbrace{\begin{pmatrix} \alpha_1 & \bullet & \bullet \\ 0 & \bullet & \bullet \\ 0 & \bullet & \bullet \\ 0 & \bullet & \bullet \end{pmatrix}}_{P_1 A} = \begin{pmatrix} \bullet & \bullet & \bullet \\ 0 & x_1^{(2)} & \bullet \\ 0 & x_2^{(2)} & \bullet \\ 0 & x_3^{(2)} & \bullet \\ 0 & x_4^{(2)} & \bullet \end{pmatrix}$$

- Matic P stačí n (počet sloupců A)
- Asymptoticky stejné jako Givensovy rotace, ale o menší koeficient
- Givensovy rotace jsou vhodné, pokud je pod diagonálou málo nul

Použití při OLS

- Hledáme argument minima ($w \in \mathbb{R}^{p+1}$)

$$\|Xw - Y\|$$

- Předpokládáme lineárně nezávislé sloupce $X \in \mathbb{R}^{N, p+1}$
- $X^T X$ je pak regulární a OLS je jednoznačně daný
- Najdeme QR rozklad X

$$X = QR = \begin{pmatrix} Q_L & Q_R \end{pmatrix} \begin{pmatrix} R_S \\ \theta \end{pmatrix}$$

- Násobení Q nemění hodnotu, protože Q je regulární
- Proto mají X a R stejnou hodnotu
- A hodnota R je $p + 1$, takže čtvercová R_S je regulární, s nenulovou diagonálou
- Dosadíme rozklad do kvadrátu normy (to hledané minimum nezmění)

$$\|Xw - Y\|^2 = \|QRw - Y\|^2 = \|Q^T(QRw - Y)\|^2 = \|Rw - Q^T Y\|^2 =$$

$$\|Rw - Q^T Y\|^2 = \left\| \begin{pmatrix} R_S w \\ \theta \end{pmatrix} - \begin{pmatrix} Q_L^T Y \\ Q_R^T Y \end{pmatrix} \right\|^2 = \left\| \begin{pmatrix} R_S w - Q_L^T Y \\ -Q_R^T Y \end{pmatrix} \right\|^2 =$$

$$\left\| \begin{pmatrix} R_S w - Q_L^T Y \\ -Q_R^T Y \end{pmatrix} \right\|^2 = \|R_S w - Q_L^T Y\|^2 + \|Q_R^T Y\|^2$$

- Druhý sčítanec nezávisí na w a lze jej tak zahodit, řešíme tedy

$$\hat{w}_{OLS} = \underset{w \in \mathbb{R}^{p+1}}{\operatorname{argmin}} \|Xw - Y\| = \underset{w \in \mathbb{R}^{p+1}}{\operatorname{argmin}} \|R_S w - Q_L^T Y\|$$

- R_S je regulární a rovnice

má proto jednoznačné řešení

- Výpočet je pak jednoduchý
 - Násobení $Q_L^T Y$ má složitost $O(N(p + 1))$
 - Řešení soustavy je díky horní trojúhelníkové R_S triviální - jen dosazujeme

QR algoritmus pro hledání vlastních čísel

- Definice

- Vlastní číslo

$$Ax = \lambda x$$
- Podobná matice
 - Čtvercové matice A, B jsou si podobné, pokud existuje regulární P t.ž.

$$A = PBP^{-1}$$
 - Je relací ekvivalence
 -
 - Diagonalizovatelná matice: podobná diagonální
 - Diagonální je tvořena vlastními čísly
 - P má ve sloupcích vlastní vektory
- Předpoklady
 - Podobnost zachovává vlastní čísla (stejný char. polynom)
 - Horní trojúhelníková matice má vlastní čísla na diagonále (determinant je součin diagonálních prvků)
 - Matice A je podobná RQ

$$A = QR \Rightarrow R = Q^T A \Rightarrow RQ = Q^T A Q$$
- Hledáme vlastní čísla čtvercové matice $A = A_0$
- Opakovaně
 - Najdeme QR rozklad

$$A_0 = Q_0 R_0$$
 - Vypočítáme podobnou matici

$$A_1 = R_0 Q_0$$
- Získali jsme posloupnost podobných matic A_0, A_1, \dots
- Posloupnost často konverguje k horní trojúhelníkové matici
- Když se prvky pod diagonálou dostatečně přiblíží nule, z diagonály získáme vlastní čísla
- Konvergenci lze zrychlit Hessenbergovou formou matice
 - Hessenbergova forma: nuly pod poddiagonálou
 - Tridiagonální forma: nuly pod poddiagonálou a nad naddiagonálou
 - Ke každé čtvercové matici existuje podobná v Hessenbergově formě
 - Pro symetrickou navíc i tridiagonální
 - Householderovými reflexemi $A^T A$ převedeme na podobnou tridiagonální B
 - Vynulujeme prvky pod poddiagonálou
 - QR algoritmus aplikujeme na $B = B_0$
 - QR rozklad v něm jednoduše uděláme $n - 1$ Givensovými rotacemi pro vynulování poddiagonály
 - Získaná $B_1 = R_0 Q_0$ zůstává v Hessenbergově formě

18 – Maticové faktorizace pomocí SVD, její výpočet, vlastnosti a použití ve strojovém učení: souvislost s metodou hlavních komponent (PCA)

Předpoklady

- $A^T A$ je pozitivně semidefinitní a symetrická
- Symetrie implikuje
 - Diagonalizovatelnost ($S = QDQ^T$, kde Q je ortogonální)
 - Reálná nezáporná vlastní čísla
 - Vlastní vektory tvoří ortonormální bázi \mathbf{R}^n
- $A^T A$ má stejnou hodnotu jako A ($\min\{m, n\}$)
- $A^T A$ má n vlastních čísel, vlastní číslo 0 má násobnost $n - r$ pro $r < n$
 - Vlastní čísla označíme $\sigma_1^2 \geq \sigma_2^2 \geq \dots \geq \sigma_r^2 \geq 0$
- Vlastním číslům přiřadíme ortonormální soubor vlastních vektorů (v_1, \dots, v_r)
 - Pro $r < n$ lze doplnit na ortonormální bázi

Výpočet

- Pro každé $i = 1 \dots r$ definujeme vektor
$$\mathbf{u}_i = \frac{1}{\sigma_i} \mathbf{A} \mathbf{v}_i$$
 - Soubor (u_1, \dots, u_r) je ortonormální (rozepteš $u_i^T u_j$)
 - Vektory u_i jsou vlastními vektory AA^T (s vlastními čísly σ_i^2)
 - AA^T a $A^T A$ mají proto stejná vlastní čísla
- Sestavíme matice rozkladu
 - V má ve sloupcích vektory v_i
 - U má ve sloupcích vektory u_i
 - Díky ortonormalitě obou souborů vlastních vektorů jsou obě matice ortogonální: $U^T U = V^T V = I_m$
 - Σ má na diagonále vlastní čísla σ_i (bez kvadrátu)
- Z definice u_i platí
$$\mathbf{A} \mathbf{v}_i = \sigma_i \mathbf{u}_i$$
a proto
$$\mathbf{A} \mathbf{V} = \mathbf{U} \Sigma,$$
odkud získáme SVD rozklad

Vlastnosti

- Rozklad matice $A \in \mathbb{R}^{m,n}$
$$A = U \Sigma V^T$$
 - $U \in \mathbb{R}^{m,m}$ ortogonální
 - $\Sigma \in \mathbb{R}^{m,n}$ diagonální s čísly σ_i
 - $V^T \in \mathbb{R}^{n,n}$ ortogonální
- Číslo σ_i jsou singulární hodnoty matice A
- Rozklad je určen jednoznačně až na
 - Znaménko vlastních vektorů

Využití - aproximace maticí nízké hodnoti

- V maticích U , Σ , V lze uvažovat jen prvních r sloupců, protože zbylé singulární hodnoty jsou nulové
 - $r = h(A)$, protože hodnota součinu nemůže být vyšší než hodnota činitele
 - Lze tak plně rekonstruovat matici A
 - Podprostor generovaný sloupci A je lineárním obalem souboru (u_1, \dots, u_r)
 - Matici A tedy reprezentujeme jako

$$A = BC, \text{ kde}$$

$$B = U_r \quad \text{a} \quad C = \Sigma_r V_r^T$$

- Můžeme ale vzít i méně než $r = h(A)$ za cenu odchylky rekonstrukce
 - Otázkou pak je, jak vzdálená bude aproximace od původní matice A
 - Vzdálenost matic měříme Frobeniovou normou
 - Frobeniova norma matice $A \in \mathbb{R}^{m,n}$ je odmocnina součtu kvadrátů prvků

$$\|A\|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n a_{ij}^2}$$

- Pro vzdálenost matic pak kvadrátů rozdílů prvků
- Odpovídá odmocnině součtu kvadrátů singulárních hodnot

$$\|A\|_F = \sqrt{\sum_{i=1}^r \sigma_i^2}$$

- Eckart-Young-Mirsky: SVD rozklad omezený na $d < r$ sloupců a řádků je nejlepším odhadem matice A měřeným Frobeniovou normou
 - Vzdálenost aproximace od původní matice dávají singul. hodnoty

$$\|A - A_d^*\|_F = \sqrt{\sum_{i=d+1}^r \sigma_i^2}$$

- Platí také pro spektrální normu

$$\|A\|_2 = \sup_{x \in \mathbb{R}^n} \frac{\|Ax\|_2}{\|x\|_2} = \sigma_1$$

- Dalším využitím je výpočet pseudoinverze matice - pro singulární X v OLS

Souvislost s PCA

- PCA se počítá pro matici $X \in \mathbb{R}^{N,p}$
 - X je vycentrovaná, průměry příznaků ve sloupcích jsou rovné 0
 - Matici vycentrujeme prostým posunutím hodnot příznaků
 - Hodnost X je r
 - Hlavní komponenty jsou dané vlastními vektory
 - Rozptyly komponent jsou vlastní čísla
- Matici kovarianci příznaků získáme z

$$\frac{1}{p-1} \mathbf{X}^T \mathbf{X}$$

- Dosadíme do kovarianční matice SVD rozklad

$$\frac{1}{p-1} \mathbf{X}^T \mathbf{X} = \frac{1}{p-1} \mathbf{V} \mathbf{\Sigma}^T \mathbf{U} \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T = \frac{1}{p-1} \mathbf{V} \mathbf{\Sigma}^T \mathbf{\Sigma} \mathbf{V}^T$$

- Získáme čtvercovou diagonální matici se spektrálním rozkladem kovarianční matice

$$\mathbf{\Sigma}^T \mathbf{\Sigma} = \begin{pmatrix} (\sigma_1)^2 & 0 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & (\sigma_2)^2 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 0 & (\sigma_3)^2 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & (\sigma_r)^2 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & \cdots & 0 \end{pmatrix} \in \mathbb{R}^{p,p}$$

- Jelikož vlastní čísla X jsou kovariance, hlavní komponenty X odpovídají prvním r sloupcům matice V
- Rozptyly hlavních komponent získáme z diagonály $\mathbf{\Sigma}^T \mathbf{\Sigma}$ vydělené $p-1$

19 – Hladká optimalizace (bez vazeb), spádové metody, volba směru a délky kroku.

Hladká optimalizace bez vazeb

- Totální derivace: lineární zobrazení $\lambda: \mathbf{R}^n \rightarrow \mathbf{R}^m$, pro které platí

$$\lim_{h \rightarrow 0} \frac{\|f(a+h) - f(a) - \lambda(h)\|}{\|h\|} = 0$$

- Pro funkci $f: D \subset \mathbf{R}^n \rightarrow \mathbf{R}^m$
- Značíme $Df(a)$
- Narozdíl od parciální derivace zohledňuje závislosti mezi proměnnými
- Pro f diferencovatelnou v bodě a
 - Existuje λ právě jedno
 - $(Df(a))_{i,j} = \partial_j f_i(a)$
- Jacobiho matice: matice $Df(a)$
- Spojitě diferencovatelná f : spojitá na otevřeném okolí bodu a , existuje $\partial_j f_i(a)$ pro všechna $1 \leq i \leq m, 1 \leq j \leq n$
 - Existuje pro ni pak $Df(a)$
- Řetězové pravidlo
 - Pro funkci g, f , kde f je diferencovatelná v $g(a) \in D_f$
 - $f \circ g$ je pak diferencovatelná v a a platí
$$D(f \circ g)(a) = Df(g(a)) \circ Dg(a)$$
 - Speciálně pro $f \circ g$ se skalárním vstupem i výstupem (výstup g a vstup f má stále libovolnou dimenzi m) pak

$$D(f \circ g)(a) = Df(g(a)) \circ Dg(a) = \nabla f(g(a))^T \cdot \begin{pmatrix} \partial g_1(a) \\ \vdots \\ \partial g_m(a) \end{pmatrix}$$

tedy součin gradientu f a derivací složek g

- Gradient
 - Směr největšího růstu
 - Derivaci ve směru získáme jako derivaci jednorozměrné funkce $h(t) = f(a + ts)$
 - Pro $g(t) = a + ts$ pak

$$Dh(t) = D(f \circ g)(t) = \nabla f(g(t))^T \cdot \begin{pmatrix} \partial g_1(t) \\ \vdots \\ \partial g_m(t) \end{pmatrix} = \nabla f(a + ts)^T \cdot \begin{pmatrix} s_1 \\ \vdots \\ s_n \end{pmatrix}$$

- A v bodě 0 získáme vzorec pro derivaci ve směru
$$\nabla_s f(a) = Dh(0) = \nabla f(a)^T \cdot s$$
- Úloha hladké optimalizace

- Hledáme

$$\operatorname{argmin}_{x \in D \subset \mathbb{R}^n} f(x)$$

pro dvakrát spojitě diferencovatelnou f

- Iteračními metodami konstruuje posloupnost aproximací x_i konvergující ke kandidátu na lokální minimum (typicky nulový gradient)
- Přístupy

- Trust region

- Na okolí bodu x_k vytvoříme aproximaci f označenou m_k
- Hledáme minimum m_k na okolí x_k

$$p_k = \operatorname{argmin}_p \{m_k(x_k + p) \mid \|p\| \leq \Delta_k\}$$

tedy směr minimalizující f

- Line search

- Následující aproximaci x_k hledáme v nějakém směru p_k

$$x_{k+1} = x_k + \alpha_k p_k$$

kde ideální volba délky kroku α_k je

$$\alpha_k = \operatorname{argmin}_{\alpha \geq 0} f(x_k + \alpha p_k)$$

nebo nějaká aproximace tohoto minima

Volba směru

- Pro line search přístup, viz výše
- Obecná volba směru

$$\nabla f(x_k)^T \cdot p_k < 0$$

- Typické volby směru jsou ve tvaru

$$p_k = -B_k^{-1} \nabla f(x_k)$$

- Metoda největšího spádu: $B_k = I$
- Newtonova metoda: $B_k = \nabla^2 f(x_k)$
- Kvazi-newtonova metoda: $B_k \approx \nabla^2 f(x_k)$
- Spádový směr: B_k pozitivně definitní

Volba délky kroku

- Ideální volbu délky kroku označíme jako minimum funkce

$$\phi(\alpha) = f(x_k + \alpha p_k) \quad \text{pro } \alpha \geq 0$$

- Armijova podmínka

$$\phi(\alpha_k) \leq \phi(0) + \alpha_k c \phi'(0)$$

kde $0 \leq c < 1$

- Nová funkční hodnota tedy leží pod přímkou určenou parametrem c
- Podmínku splní i velmi malý krok α_k
- Algoritmus: začínáme velkým krokem, zmenšujeme koeficientem < 1 , dokud α_k nesplní podmínku (můžeme ale minout optimum)

- Goldsteinova podmínka

$$\phi(0) + \alpha_k(1 - c)\phi'(0) \leq \phi(\alpha_k) \leq \phi(0) + \alpha_k c\phi'(0)$$
 kde $0 < c < 1/2$
 - Nová funkční hodnota tedy leží mezi přímkami
 - Řešíme tím problém příliš malého kroku
 - Opět ale můžeme minout optimum
- Wolfeho podmínky
 - Slabá podmínka

$$\phi(\alpha_k) \leq \phi(0) + \alpha_k c_1 \phi'(0)$$

$$\phi'(\alpha_k) \geq c_2 \phi'(0)$$
 kde $0 < c_1 < c_2 < 1$
 - První podmínka: nová hodnota je pod přímkou
 - Druhá podmínka: sklon v novém bodě musí být větší než původní, zmírněný parametrem c_2
 - Silná podmínka

$$\phi(\alpha_k) \leq \phi(0) + \alpha_k c_1 \phi'(0)$$

$$|\phi'(\alpha_k)| \leq c_2 |\phi'(0)|$$
 kde $0 < c_1 < c_2 < 1$
 - První podmínka stejná
 - Druhá podmínka: sklon v novém bodě musí být absolutně nižší (tedy vodorovnější), čímž se vyhneme extrémům
- Pro spojitě diferencovatelnou zdola omezenou f lze vždy Wolfeho podmínky splnit

Spádové metody

- Metoda největšího spádu
 - Iterativní optimalizace typicky pro funkce s mnoha proměnnými
 - Směr kroku $p_k = -\nabla f(x_k)$ (proti směru gradientu)
 - Délka kroku podle některé z podmínek výše
 - Pokud konvergujeme k x_k^* , pro které je Hessián v bodě pozitivně definitní, víme, jak rychle metoda konverguje
 - Vzdálenost $k+1$ tého kroku od optima je shora omezena číslem závislým na největším a nejmenším vlastním čísle Hessiánu
- Newtonova metoda
 - Směr kroku p_k získáme jako $\arg\min m_k(p)$, kde m_k je aproximací f pomocí Taylorova rozvoje, tj.

$$f(x_k + p) \approx f(x_k) + \nabla f(x_k)^T p + \frac{1}{2} p^T \nabla^2 f(x_k) p = m_k(p)$$

- Parciálním derivováním získáme z $\nabla m_k(p)$ hodnotu p_k jako

$$p_k = -(\nabla^2 f(x_k))^{-1} \nabla f(x_k) \quad (\text{Hessián je } B_k \text{ z volby směru})$$

- Metoda konverguje kvadraticky rychle, tj.

$$\lim_{k \rightarrow +\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^2} < L \quad (\text{počet platných číslic se krokem zdvojnásobí})$$

za podmínek

- Funkce f je dvakrát spojitě diferencovatelná
- Gradient f je v optimu nulový a Hessián pozitivně definitní
- Hessián je Lipschitzovsky spojitý ("příliš rychle se nemění")
- x_0 je dostatečně blízko optimu
- Pokud Hessián nemá inverzi, lze přičíst kladnou diagonální matici, použít rozklad...)
- BGFS
 - Výpočet B_{k+1} aktualizací B_k
- Stochastický gradientní sestup
 - Aktualizace parametrů modelu strojového učení pomocí jednoho vzorku namísto celého datasetu najednou
 - Rychlejší varianta gradientního sestupu
 - Minimalizujeme funkci ve tvaru

$$f(x) = \frac{1}{n} \sum_{i=1}^n f_i(x)$$

tedy průměr nějaké ztrátové funkce f pro všechny datové body

- Aproximujeme x_0 a iterujeme podle

$$x_{k+1} = x_k - \eta \nabla f_{i_k}(x_k)$$

tedy proti směru gradientu škálovaným learning rate

- Modifikace s přidáním setrvačnosti

- Adagrad

$$s_0 = 0,$$

$$s_k = s_{k-1} + (\nabla f_{i_k}(x_k))^2,$$

$$x_{k+1} = x_k - \eta \frac{\nabla f_{i_k}(x_k)}{\sqrt{s_k + \varepsilon}}$$

- Momentová metoda

$$x_{k+1} = x_k - (\eta \nabla f_{i_k}(x_k) + \gamma(x_k - x_{k-1}))$$

s faktorem zapomínání < 1

20 – Časové řady: aditivní a multiplikativní dekompozice, momenty (střední hodnota, rozptyl, autokovariance). Druhy stacionarity a rozdíl mezi nimi. Základní vlastnosti náhodné procházky a bílého šumu.

Časová řada

- Časová řada: soubor pozorování x_t získaných v časových okamžicích t
 - Množina $\{x_t \mid t \in T\}$
 - Pro $T = \mathbf{N}$ jde o řadu s diskrétním časem
- Charakteristika vývoje
 - Trend: dlouhodobý vývoj střední hodnoty
 - Sezónnost: periodicky opakující se vývoj (z ACF po odebrání trendu)
 - Cyklické změny: nepravidelné fluktuace, např. ekonomické cykly

Aditivní a multiplikativní dekompozice

- Složky časové řady
 - T_t : trend
 - S_t : sezónní složka
 - E_t : nevysvětlená složka
- Modely
 - Aditivní dekompozice
$$Y_t = T_t + S_t + E_t$$
 - Amplituda sezónních složek je přibližně stejná
 - Multiplikativní dekompozice
$$Y_t = T_t \times S_t \times E_t$$
 - S rostoucím trendem se zvětšuje amplituda sezónnosti
- Výběr modelu
 - Minimalizace součet čtverců autokorelační funkce reziduí E_t
 - ACF: lineární korelace hodnot časové řady v různých posunech
 - Umožňuje odhalit sezónnost
 - Rezidua říkají, jaká míra korelace v E_t zbyla, chceme vysvětlit maximum

Momenty

- Náhodný proces: posloupnost náhodných veličin x_t pro t z indexové množiny
- Střední hodnota $E x_t$
- Variance $\sigma_t^2 = \text{var} x_t$
- Autokovariance $\gamma(t_1, t_2) = \text{cov}(x_{t_1}, x_{t_2}) = E[(x_{t_1} - \mu_{t_1}) \cdot (x_{t_2} - \mu_{t_2})]$

Stacionarita

- Definice
 - Silná stacionarita: sdružená distribuce $x_{t_1} \dots x_{t_n}$ je stejná jako $x_{t_1 + \tau} \dots x_{t_n + \tau}$ pro všechna $t_1 \dots t_n, \tau$
 - Libovolný posun tedy nemá vliv na distribuci
 - Slabá stacionarita: vůči posunům jsou invariantní momenty do druhého řádu (střední hodnota, autokovariance)
 - Ze silné nevyplývá slabá (momenty musí existovat, Cauchy)
- Typy procesů podle stacionarity
 - Stacionární: bez trendu a jednotkových kořenů, silně či slabě
 - Trend-stacionární: stacionární po odstranění trendu
 - Stacionární po diferencování (má jednotkový kořen)
 - Nestacionární
- Testy stacionarity
 - ADF (Augmented Dickey-Fuller)
 - H_0 : proces má jednotkový kořen
 - H_A : proces má kořeny vně jednotkové kružnice
 - KPSS (Kwiatkowski-Phillips-Schmidt-Shin)
 - H_0 : proces je trend-stacionární
 - H_A : proces má jednotkový kořen
 - Ani jeden nezamítne: možná trend-stacionární, nevíme
 - Oba zamítnou: heteroskedasticita, strukturální změna

Náhodná procházka a bílý šum

- Bílý šum
 - $Ex_t = 0$
 - $\text{var}(x_t) = \sigma^2 < \infty$
 - $\text{cov}(x_t, x_{t+\tau}) = \gamma(\tau) = 0$
 - Speciálním případem je gaussovský bílý šum
 - Zvuková syntéza, RNG
- Náhodná procházka
 - Mějme diskrétní bílý šum $z_t \sim L(0, \sigma^2)$
 - Proces
 - $x_0 = 0$
 - $x_t = x_{t-1} + z_t$
 - Tedy $x_t = \sum_{i=1}^t z_i$
 - $Ex_t = 0$
 - $\text{var}(x_t) = t\sigma^2$

21 – Autoregresní modely (AR) a modely klouzavých průměrů (MA): základní vlastnosti modelů/procesů, jejich stacionarita. Zápis AR a MA, včetně zápisu pomocí operátoru zpoždění. Identifikace řádů AR a MA z autokorelačních funkcí a pomocí informačních kritérií.

Autokorelační funkce

- Lineární (Pearsonův) korelační koeficient

$$\rho_{XY} = \frac{\mathbb{E}[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y} = \frac{\sigma_{XY}}{\sigma_X \sigma_Y}, \quad \rho \in [-1, 1]$$

- Pokud jsou veličiny nezávislé, pak jsou i nekorelované
- Výběrový z výběrových variancí
- ACF (Autokorelační funkce)

$$R(s, t) = \frac{\mathbb{E}[(X_t - \mu_t)(X_s - \mu_s)]}{\sigma_t \sigma_s}, \quad R(s, t) \in [-1, 1]$$

- Lineární korelace hodnot časové řady v různých okamžicích
- Pro slabě stacionární proces závisí pouze na rozdílu $s - t$

$$R(s, t) = R(s - t) = R(\tau)$$

- Parciální korelační koeficient
 - Lineárně závislé náhodné veličiny X, Y
 - Obě jsou ovlivněny třetí veličinou Z
 - Pro měření korelace X a Y je nutné je očistit od vlivu Z
 - Najdeme regresní přímky

$$\hat{X} = a + b^T Z$$

$$\hat{Y} = c + d^T Z$$

které jsou nejlepším lineárním přiblížením k X a Y pomocí Z

- Veličiny

$$e_X = (X - \hat{X}) \text{ a } e_Y = (Y - \hat{Y})$$

jsou pak rezidua, která v X a Y nedokázala Z vysvětlit, tedy jsou od vlivu Z očištěna

- Korelační koeficient mezi X a Y při daném Z : $\rho_{XY \cdot Z}$
- PACF (Parciální autokorelační funkce)

$$\alpha(k) = \rho_{X_{t+k} X_t \cdot \{X_{t+1}, \dots, X_{t+k-1}\}}$$

- Autokorelace mezi X_t a X_{t+k} s odstraněním vlivu mezilehlých hodnot

Informační kritéria

- AIC (Akaike Information Criterion) = $2k - 2\ln L$
 - k : počet odhadovaných parametrů
 - L : maximální hodnota věrohodnosti při daném modelu
- BIC (Bayesian Information Criterion) = $k\ln(n) - 2\ln L$
 - Navíc n : počet pozorování
- Informační kritéria minimalizujeme

Operátor zpoždění

$$\begin{aligned}BX_t &= X_{t-1} \\ B^{-1}X_t &= X_{t+1} \\ B^k X_t &= \underbrace{B \cdot B \cdots B}_{k \times} X_t = X_{t-k}\end{aligned}$$

Autoregresní modely

- Popis modelu na základě předchozích realizací
- AR(p) model

$$X_t = c + \phi_1 X_{t-1} + \dots + \phi_p X_{t-p} + \varepsilon_t$$

- p může být libovolné, ale musí dávat smysl
 - ε_t je bílý šum
- Charakteristický polynom

$$1 - \phi_1 z - \phi_2 z^2 - \dots - \phi_p z^p = 0$$

- Proces je stacionární, pokud její kořeny leží vně jednotkové kružnice
 - Stacionární AR proces je invertibilní na MA(∞)
- Odhad parametrů pomocí OLS

$$Y = X\phi + \epsilon$$

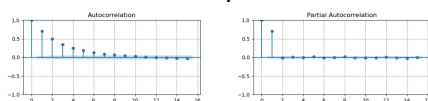
$$\hat{\phi} = (X^T X)^{-1} X^T Y$$

- Zápis pomocí operátoru zpoždění

$$X_t = \sum_{i=1}^p \phi_i X_{t-i} + \varepsilon_t \quad (\text{vyjádříme } \varepsilon_t)$$

$$\varepsilon_t = X_t - \phi_1 B^1 X_t - \dots - \phi_p B^p X_t = \left(1 - \sum_{i=1}^p \phi_i B^i\right) X_t$$

- Identifikace řádu pomocí ACF a PACF: v případě náhodné procházky s bílým šumem



- ACF postupně klesá, protože x_t je přímo ovlivněno x_{t-1} atd.
 - PACF indikuje silnou korelaci mezi x_t a x_{t-1} , ale zbytek už je očištěn

Modely klouzavých průměrů

- Propagace šumové složky, nebere v potaz předchozí hodnoty
- MA(q) model

$$X_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \dots + \theta_q \varepsilon_{t-q}, \quad \varepsilon_t \sim \text{iid } \mathcal{N}(0, \sigma^2)$$

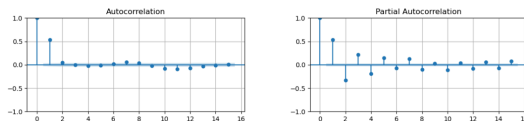
- Náhodné šoky bílého šumu, tedy i.i.d.
- Vlastnosti

$$\mathbb{E}[X_t] = c, \quad \text{var } X_t = \sigma^2 \cdot \sum_{i=0}^q \theta_i^2$$

- Invertibilita
 - Zajímá nás místo stacionarity
 - Proces je invertibilní, má-li kořeny charakteristického polynomu vně jednotkové kružnice
 - Invertibilní MA proces lze převést na AR(∞)
 - PACF MA má proto mnoho významných lagů
 - MA je vždy stacionární, protože vznikne ze stacionárního AR
- Odhad numerickou optimalizací
- Zápis pomocí operátoru zpoždění

$$X_t = \left(1 + \sum_{i=1}^q \theta_i B^i\right) \varepsilon_t$$

- Identifikace řádu pomocí ACF a PACF: v případě MA(1) bílým šumem:



- ACF vysoká pro $k = 1$, zbytek nulový, protože x_t nezávisí na x_{t-1} atd. díky nekorelovanému šumu
- PACF bude postupně klesat k 0

22 – Smíšené modely ARIMA: základní vlastnosti modelů/procesů, integrování a diferencování. Zápis ARIMA, včetně zápisu pomocí operátorů zpoždění a difference, speciální případy podle hodnot p , d , q . Problém redundance parametrů.

Smíšený model ARMA(p , q)

- Standardní zápis

$$X_t = c + \phi_1 X_{t-1} + \dots + \phi_p X_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \dots + \theta_q \varepsilon_{t-q}$$

- Pomocí backshift operátoru

$$\Phi X_t = \Theta \varepsilon_t, \quad \text{kde} \quad \Phi = 1 - \sum_{i=1}^p \phi_i B^i \quad \text{a} \quad \Theta = 1 + \sum_{i=1}^q \theta_i B^i$$

- Normální šum
 - Flexibilní model, ale mnoho neznámých
 - Ekonometrie: AR vývoj podle minulosti + MA šokové změny
 - Předpoklad slabé stacionarity
 - Zajistíme odstraněním trendu
 - Nebo diferencí - ARIMA
- $$X'_t = X_t - X_{t-1}$$
- Potřebu diferenciacce poznáme z velmi pomalu klesající ACF
 - Box-Jenkins přístup k ARMA
 - Identifikace modelu: odhalení řádu AR, MA z ACF, PACF
 - Transformace pro zajištění stacionarity
 - Odhad parametrů numericky
 - Ověření modelu: nekorelovanost reziduí a jejich slabá stacionarita

Smíšený model ARIMA(p , d , q)

- Operátor difference

$$\nabla X_t = X_t - X_{t-1} = (1 - B)X_t,$$

$$\begin{aligned} \nabla^2 X_t &= \nabla(\nabla X_t) = \nabla X_t - \nabla X_{t-1} = (1 - B)\nabla X_t \\ &= (1 - B)^2 X_t, \end{aligned}$$

$$\nabla^k X_t = (1 - B)^k X_t.$$

- Definujeme ARIMA(p , d , q)

$$\Phi(1 - B)^d X_t = \Theta \varepsilon_t, \quad \text{kde} \quad \Phi = 1 - \sum_{i=1}^p \phi_i B^i \quad \text{a} \quad \Theta = 1 + \sum_{i=1}^q \theta_i B^i$$

- S operátory se dobře počítá a snadno se zapíše char. Polynom

- Časové řady s trendem nebo náhodnou procházkou jsou silně pozitivně autokorelované
 - ACF s velkými korelacemi
 - PACF typicky blízko 1 v prvním lagu
- Volba d
 - $d = 1$: konstantní průměrný trend (náhodná procházka s driftem)
 - Vyšší d málokdy, nepoužíváme při něm konstantu c v modelu
 - AR charakteristika může značit poddiferencovanost
 - MA charakteristika může značit přediferencovanost
- Role konstanty
 - $d = 0$: zavádí nenulovou střední hodnotu, vyplatí se zkusit
 - $d = 1$: zavádí nenulový průměrný trend, může se hodit
 - $d = 2$: význam "trendu v trendu", obvykle nechceme
- Obecná pravidla
 - Rádi bychom $p = 0$ nebo $q = 0$ (ve fyzice často $q = p - 1$)
 - $p + q \leq 3$
 - Redundance parametrů: AR a MA části modelu se mohou vzájemně vyrušovat a zvyšuje se složitost
- Běžné ARIMA modely

ARIMA (0, 0, 0) + c	Konstantní model
ARIMA (0, 1, 0)	Model náhodné procházky
ARIMA (0, 1, 0) + c	Náhodná procházka s driftem
ARIMA (1, 0, 0) + c	AR (1)
ARIMA (2, 0, 0) + c	AR (2)
ARIMA (1, 1, 0) + c	AR (1) na 1x diferencovaných datech
ARIMA (2, 1, 0) + c	AR (2) na 1x diferencovaných datech
ARIMA (0, 1, 1)	Jednoduché exponenciální vyhlazování = MA (1) na 1x dif. Datech
ARIMA (0, 1, 1) + c	MA (1) na 1x diferencovaných datech s konstantním lineárním trendem
ARIMA (1, 1, 2)	Lineární exponenciální vyhlazování s tlumeným trendem
ARIMA (0, 2, 2)	Zobecněné lineární exponenciální vyhlazování
ARIMA (1, 0, 0) + c	Podle toho, jaké parametry má drift c : <ol style="list-style-type: none"> 1. $\Phi_1 = 0 \rightarrow$ bílý šum 2. $\Phi_1 = 0, c = 0 \rightarrow$ náhodná procházka 3. $\Phi_1 = 0, c \neq 0 \rightarrow$ náhodná procházka s driftem 4. $\Phi_1 < 0 \rightarrow$ řada oscilující mezi klad. a záp. hodnotami