Okruhy tém a otázky na skúšku zo Strojového učenia

1. Rámec strojového učenia, základné pojmy
2. Priestor pojmov, príznakov a stavový priestor
3. Reprezentácia vstupov a výstupov

---

4. Klasifikačná, regresná a sekvenčná úloha

Klasifikačná, regresná a sekvenčná úloha sú typy úloh v oblasti strojového učenia. Každá z týchto úloh má iné ciele a požiadavky na model, ktorý sa snaží riešiť danú úlohu.

- Klasifikačná úloha sa zameriava na rozdelenie vstupných dát do jednej alebo viacerých preddefinovaných tried. Modely využívané pre klasifikáciu sa často označujú ako klasifikátory a môžu používať rôzne algoritmy na identifikáciu vzorov v dátach, ktoré by mohli naznačovať, do ktorej triedy daný vstupný dát patrí. Príklady klasifikačných úloh sú napríklad rozpoznávanie rukopisu, identifikácia spamu v e-mailoch alebo rozpoznávanie tvárí.

- Regresná úloha sa zameriava na predpovedanie numerických hodnôt na základe vstupných dát. Modely využívané pre regresiu môžu používať rôzne algoritmy, ktoré sa snažia identifikovať vzťah medzi vstupnými a výstupnými dátami a tým predpovedať hodnotu výstupu pre nové vstupné dáta. Príklady regresných úloh sú napríklad predpovedanie cien nehnuteľností, odhadovanie pravdepodobnosti úrazu v závislosti na rôznych faktoroch alebo predpovedanie ceny akcií na burze.

- Sekvenčná úloha sa zameriava na prácu s časovo závislými dátami, kde každá hodnota vstupu a výstupu závisí od predchádzajúcich hodnôt. Modely využívané pre sekvenčné úlohy musia byť schopné udržiavať kontext a pamäť o predchádzajúcich dátach a na základe toho predpovedať budúce hodnoty. Príklady sekvenčných úloh sú napríklad rozpoznávanie reči, predpovedanie pohybu na základe časovo závislých senzorových dát alebo predpovedanie textu na základe jeho predchádzajúceho kontextu.

Je dôležité poznamenať, že tieto kategórie úloh nie sú vzájomne vylučujúce a v praxi sa môžu používať aj kombinácie

5. Delenie algoritmov SU

Algoritmy strojového učenia sa môžu rozdeliť na tri kategórie v závislosti na ich spôsobe učenia a použitia vstupných dát. Tieto kategórie sú:

1. Supervízované učenie: Algoritmy založené na supervízovanom učení sa učia na základe dátovej sady obsahujúcej páry vstupných a výstupných dát. Ich cieľom je vytvoriť funkciu, ktorá môže predikovať výstupnú hodnotu pre nové vstupné dáta. Medzi príklady algoritmov založených na supervízovanom učení patria klasifikačné a regresné modely.

2. Nesupervízované učenie: Algoritmy založené na nesupervízovanom učení sa učia na základe dátovej sady, ktorá neobsahuje žiadne označené výstupy. Ich cieľom je identifikovať štruktúry, vzory a závislosti v dátach bez konkrétneho účelu predikcie výstupov. Medzi príklady algoritmov založených na nesupervízovanom učení patria klastrovacie algoritmy alebo redukcia dimenzionality.

3. Zosilňovanie učenia: Algoritmy založené na zosilňovaní učenia sa učia na základe interakcie s prostredím a získavajú spätnú väzbu na základe toho, ako dobre sa im darí riešiť úlohu. Ich cieľom je optimalizovať rozhodovacie procesy tak, aby dosiahli maximálny výkon v danom prostredí. Medzi príklady algoritmov založených na zosilňovaní učenia patria napríklad hry so zosilňovaním učenia alebo robotické systémy.

Delenie algoritmov strojového učenia na tieto kategórie je užitočné pre lepšie porozumenie toho, aké typy algoritmov sa používajú na rôzne úlohy a aké sú ich hlavné vlastnosti. Tieto kategórie algoritmov môžu byť tiež využité pri výbere najvhodnejšieho algoritmu pre konkrétnu úlohu v závislosti na typu dát a požiadavkách na výkon a presnosť.

6. Komplexnosť učenia, preferencie modelov (bias)

Strojové učenie má niekoľko rôznych aspektov, ktoré ovplyvňujú úspešnosť učenia a správanie sa modelov. Dva z týchto aspektov sú komplexnosť učenia a preferencie modelov (bias).

1. Komplexnosť učenia: Táto vlastnosť sa týka schopnosti algoritmu strojového učenia naučiť sa zložité vzťahy v dátach. Algoritmy s nízkou komplexnosťou sa môžu naučiť jednoduché vzťahy v dátach, zatiaľ čo algoritmy s vysokou komplexnosťou môžu naučiť sa zložité vzťahy a prejavy v dátach. Avšak algoritmy s vysokou komplexnosťou môžu byť náchylné na preučenie (overfitting), čo znamená, že sa naučia príliš presne závislosti v trénovacej sade a nemusia dobre generalizovať na nové dáta.

2. Preferencie modelov (bias): Táto vlastnosť sa týka predpokladov, ktoré sú vstavané do algoritmu strojového učenia. Tieto predpoklady môžu ovplyvniť to, aký typ modelu bude algoritmus preferovať pri riešení danej úlohy. Napríklad lineárne regresné modely majú predpoklad, že výstupná premenná závisí lineárne od vstupných premenných, takže algoritmy založené na lineárnej regresii budú preferovať modely, ktoré sa dobre hodia k tomuto predpokladu. Tieto preferencie modelov môžu byť užitočné, ak algoritmus učenia vie, že daný predpoklad zodpovedá úlohe, ale môžu byť obmedzujúce, ak predpoklad nie je správny.

Rozumieť komplexnosti učenia a preferenciám modelov (bias) je dôležité pre lepšie porozumenie správania sa algoritmov strojového učenia a zlepšenie výsledkov učenia. Pri výbere algoritmu a modelu je dôležité zvážiť aj tieto vlastnosti a zvoliť najvhodnejšiu kombináciu pre danú úlohu.

---

7. Generovanie konjunkcií, reprezentácia a použitie

8. Inkrementálna indukcia logických konjunkcií prehľadávaním priestoru pojmov
9. VSS algoritmy (SG, GS a obojsmerné)
10. Neinkrementálna indukcia logických konjunkcií
11. Úplné prehľadávanie (Exhaustive - EGS, ESG)
12. Heuristické prehľadávanie (Heuristic HGS, HSG)
13. Indukcia klasifikačných pravidiel, reprezentácia a použitie
14. Neinkrementálna aplikácia princípu rozdeľ a panuj (NSC)
15. Generovanie klasifikačných pravidiel algoritmami AQ (AQ11)

---

16. Generovanie rozhodovacích stromov klasifikačného typu

o Každý rozhodovací stom RS je možné transformovať na **sadu neusporiadaných pravidiel**

o RS je možné použiť na **nepriame generovanie klasifikačných pravidiel**

o Algoritmy, ktoré generujú na výstupe Rozhodovací Strom (RS)pracujú väčšinou neinkrementálne

o Uplatňujú princíp **„rozdeľuj a panuj“** resp. princíp „**delenia priestoru príkladov na pod-priestory**“

o Ukončovacia podmienka (UK)

o Pre **perfektnú klasifikáciu: každý pod-priestor obsahuje iba príklady jednej a tej istej triedy**

o Pre nie perfektnú klasifikáciu: každý pod-priestor obsahuje aspoň dané hraničné percento príkladov jednej triedy (napr. 90%) – generovanie RS nevykoľají prítomnosť zašumených dát

Všeobecný popis algoritmu na generovanie RS:

Ak je pre každý pod-priestor splnená UK à Potom KONIEC

Inak 1.Zvoľ pod-priestor obsahujúci príklady rôznych tried

2.Zvoľ preň ešte nepoužitý testovací atribút (TA)

3.Rozdeľ ho na ďalšie pod-priestory podľa hodnôt TA

---

17. Algoritmy ID3 a ID5R

Algoritmus ID3 – kategorické dáta

o je **neinkrementálny** algoritmus

o Generuje **klasifikačný model** - uskutočňuje **perfektnú klasifikáciu**, lebo algoritmus používa UK, kým pod-priestory neobsahujú iba TP jednej triedy

o Generuje model v tvare **minimálneho stromu**, lebo vyberá TA pomocou **Shannonovej teórie informácie**

Základom je výpočet **entrópie**, teda **neurčitosti priestoru príkladov** – vzorec pre dve triedy:

$$H(S) = -\sum_{j=1}^{2} \frac{n_j}{n_1 + n_2} \log_2 \frac{n_j}{n_1 + n_2}$$

Shannonová teória informácie

Výpočet zmeny **entrópie**, po rozdelení priestoru príkladov podľa hodnôt TA, kde H je počet hodnôt atribútu – **H(S,A) je váhovaná entrópia dcérskych uzlov**

o Algoritmus generuje model v tvare **minimálneho stromu**, za predpokladu splnenia nasledovných podmienok:

- o Podmienky **neprotirečivosti** resp. nekontradikčnosti TP
- o Podmienky **ne-redundantnost**i TP
- o Podmienky vzájomne **nezávislosti** atribútov

Algoritmus ID5R - kategorické dáta

o Je **inkrementálny** algoritmus, ktorý generuje **klasifikačný model**

o Je modifikáciou ID3

o Po každom novom príklade aktualizuje model - **modifikuje existujúci strom** - negeneruje celý RS odznova

o **Maximalizuje informačný zisk** ako ID3

o V každom uzle RS uchováva všetky informácie počtoch TP – potrebné na overenie vhodnosti TA po zmene pomerov

o Po zmene TA **reštrukturalizuje celý pod-strom**

---

18. Vylepšenia algoritmu C4.5 (pomerové kritérium zisku, spojité atribúty)

Pomerové kritérium zisku

o Na rozdiel od ID3 neuprednostňuje TA s väčším počtom hodnôt

o Normalizuje informačný zisk pomerovou entrópiou

o Pomerová entrópia:

o narastá s rastúcim počtom vetiev à vyššia penalizácia (pôvodný informačný zisk IG je delený vyššou hodnotou)

o vytvára novú **prehľadávaciu preferenciu**

o vytvára **rozmerovo menší RS** (do šírky)

Spracovanie spojitých atribútov

o TP sú usporiadané podľa hodnôt spojitého atribútu

o Formuje sa množina usporiadaných hodnôt, v ktorej sa každá hodnota vyskytuje iba raz

o Nájde sa **prahová hodnota** pre každú susednú dvojicu hodnôt spojitého atribútu, ktorá rozdelí TP na dve množiny - binárne delenie

o Ide o **transformáciu spojitého atribútu na binárny atribút**

o Prahová hodnota sa určí ako a**ritmetický priemer daných dvoch susedných hodnôt**

o Pre m hodnôt atribútu existuje m-1 rozdelení – prahových hodnôt

o Pre rozdelenia sa vypočíta pomerový informačný zisk

o **Vyberie sa jediné rozdelenie TP podľa prahovej hodnoty s maximálnym pomerovým informačným ziskom**

o Priestor TP sa rozdelí na dva pod-priestory podľa zvoleného prahu

o Ten istý spojitý atribút sa môže použiť pri generovaní stromu znova ale s novou ešte nepoužitou prahovou hodnotou

---

19. Rozhodovacie stromy regresného typu – reprezentácia, použitie

o Algoritmy na generovanie RS sa môžu aplikovať aj na regresnú úlohu. Potom generujú regresný strom

o Regresný RS tiež reprezentuje rozhodovaciu procedúru

o Aj tu sa uplatňuje princíp „rozdeľuj a panuj“ resp. princíp „delenia priestoru príkladov na pod-priestory“

o Aplikuje sa na **numerické dáta** (nie kategorické)

o **Segmentuje priestor príznakov, atribútov** (predictors space) na určitý počet jednoduchých oblastí

o Regresný strom predikuje numerickú hodnotu pre nové pozorovanie z ktorého sa neučil

Použitie:

Novému TP je predikovaná presná hodnota oblasti v priestore príznakov do ktorej TP patrí na základe jeho hodnôt atribútov.

Príklady použitia:

Regresný strom – predikcia celočíselnej hodnoty

· Koľko Jano zaplatí za svoj paušál budúci rok? (€)

· Koľko paušálov predá Orange budúci rok? (#)

---

20. Generovanie regresných stromov (CART)
    - Neinkrementálny, neparametrická metóda
    - Samo-validačný (train/validation/test)
    - Na výber testovacích atribútov – prediktorov používa GINI index

$$Gini = 1 - \sum_j p_j^2 \qquad Entropy = -\sum_j p_j \log_2 p_j$$

verzus

    - Realizuje binárne delenie (binárny split) - Strom rastie do hĺbky
    - Výsledky sú invariantné lineárnym transformáciám atribútov

    napr. x'=Log(x) - nie je citlivý na extrémne hodnoty

    - Samo-orezávanie (minimálna chyba na validačnej množine)
    - Nestabilný, malá zmena v dátach môže znamenať, že sa vygeneruje celkom iný regresný strom

---

21. Stromy verzus lineárne modely

Vhodnosť použitia závisí na probléme teda na charaktere dát

    - Ak závislosť výstupu (odpovede, predikcie) na hodnotách atribútov (prediktorov) má lineárny charakter, teda máme lineárne dáta, potom sú vhodné lineárne modely:
        - Lineárna prahová jednotka,
        - Lineárne SVM
        - Lineárna regresia
    - Ak je táto závislosť komplexná a vysoko nelineárna, teda máme nelineárne dáta, potom sú vhodné stromové modely:
        - Klasifikačný rozhodovací strom
        - Regresný strom

Náhodné lesy (Random Forests)

---

22. Rozhodovací zoznam ako usporiadaný systém pravidiel

---

23. Generovanie rozhodovacích zoznamov (algoritmus NEX)

    Indukuje Rozhodovací Zoznam (RZ) neinkrementálne použitím výnimiek
    Generuje usporiadaný súbor pravidiel

    Všeobecný popis algoritmu:

    1. Formuje implicitné pravidlo pre najfrekventovanejšiu triedu
    2. V každej iterácii aplikuje naučený model (RZ) na všetky TP
    3. Chybne klasifikované príklady zaradí medzi výnimky
    4. Pre výnimky sformuje nové pravidlo a pridá ho na začiatok RZ
    5. Ukončovacia podmienka: algoritmus korektne klasifikoval všetky TP pomocou aktuálneho RZ

---

24. Numerické charakteristiky pravidla (algoritmus CN2)
25. Prahové pojmy – reprezentácia a použitie
26. Generovanie pojmu n_of_m tabuľkou kritérií (algoritmus HCT)
27. Naučenie lineárnej a sférickej prahovej jednotky

---

28. Iteratívna perturbácia váh lineárnej deliacej hyperroviny (algoritmus IWP)

    IWP - Iterative Weight Perturbation

    - teda Iteratívna váhová perturbácia

    Vstupy: ISET...množina trénovacích príkladov

    ATTS...množina atribútov

    Parameter: Max_Iterations...maximálny počet iterácií

    Výstupy: LTU na klasifikáciu nových príkladov

    Procedúra: iwp (ISET,ATTS).

    Inicializácia:

    nech H je LTU s voliteľnými váhami z intervalu (-1,1)

    nech BEST = H

    nech COUNT = Max_Iterations

    Iterative Weight Perturbation (IWP) je metóda, ktorá používa iteratívny proces na úpravu váh Lineárnej transformačnej jednotky (LTU) s cieľom optimalizovať jej schopnosť klasifikovať nové príklady. Tu je základný rámec toho, ako by mohol tento algoritmus fungovať:

    1. **Inicializácia**: Vytvoríte LTU s váhami, ktoré sú náhodne inicializované z intervalu (-1,1). Táto LTU sa označí ako "BEST". Taktiež nastavíte počítadlo "COUNT" na maximálny počet iterácií.

2. **Iterácia**: Nasledujúcich krokov sa bude opakovať až do dosiahnutia maximálneho počtu iterácií:

- Vypočítajte chybu aktuálnej LTU na trénovacej množine.

- Náhodne perturbujte váhy LTU a vypočítajte chybu na trénovacej množine s novými váhami.

- Ak je chyba s novými váhami menšia ako chyba s pôvodnými váhami, aktualizujte "BEST" na aktuálnu LTU a jej váhy.

- Aktualizujte "COUNT" znížením o jedna.

3. **Výstup**: Po ukončení iterácií vráti algoritmus "BEST" ako výslednú LTU, ktorú môžete použiť na klasifikáciu nových príkladov.

---

29. Od lineárnej prahovej jednotky ku SVM

Lineárna prahová jednotka (LTU) a podporný vektorový stroj (SVM) sú dva typy algoritmov používaných v strojovom učení na klasifikáciu a regresiu. Hoci obe techniky sa snažia nájsť rozhodovacie hranice, ktoré oddelia dáta, majú niekoľko kľúčových rozdielov.

1. **Lineárna prahová jednotka** (LTU): LTU je najjednoduchší typ umelého neurónu a môže byť použitý na vytvorenie lineárnej klasifikácie. LTU prijíma vstupy, váži ich pomocou sady váh, a ak je súčet týchto vážených vstupov väčší ako určitý prah, výstup je aktívny (často sa označuje ako 1); inak je neaktívny (často sa označuje ako 0). LTU je teda schopný vytvoriť len lineárne rozhodovacie hranice.

2. **Podporný vektorový stroj** (SVM): SVM je výkonnejší algoritmus klasifikácie a regresie. Pri klasifikácii sa SVM snaží nájsť hyperrovinu, ktorá najlepšie oddelí dve triedy s maximálnou maržou, čo znamená, že sa snaží maximalizovať vzdialenosť medzi najbližšími príkladmi každej triedy (podpornými vektormi) a hyperrovinou. SVM môže taktiež vytvárať nelineárne rozhodovacie hranice prostredníctvom techniky nazývanej kernel trick, ktorá transformuje vstupné dáta do vyššieho dimenzionálneho priestoru, kde sú dáta lineárne separovateľné.

Prechod z LTU na SVM by teda mohol zahŕňať nasledujúce kroky:

1. Požiadať o dáta: Oba algoritmy vyžadujú súbor dát na trénovanie.

2. Predspracovanie dát: Oba algoritmy môžu vyžadovať normalizáciu alebo iné formy predspracovania dát.

3. Trénovanie modelu: LTU sa trénuje prispôsobením váh na základe chyby medzi predpoveďou a skutočnou hodnotou, zatiaľ čo SVM sa trénuje hľadaním hyperroviny, ktorá maximalizuje maržu medzi triedami.

4. Testovanie a validácia modelu: Po trénovaní sa oba modely testujú a validujú na základe ich schopnosti správne klasifikovať nové dáta.

---

30. SVM – klasifikátor maximálneho rozpätia

Podporné vektorové stroje (SVM - Support Vector Machines) sú silné algoritmy používané v strojovom učení, najmä pre úlohy klasifikácie a regresie. Koncept "maximálneho rozpätia" (alebo "maximálnej marže") je kľúčový pre fungovanie SVM.

V kontexte binárnej klasifikácie, SVM funguje tak, že nájde hyperrovinu v priestore vstupných premenných, ktorá oddelí dva typy bodov (reprezentujúcich dve triedy) s čo najväčším rozpätím alebo maržou. Hyperrovina je definovaná tak, že vzdialenosť (marža) medzi najbližšími bodmi každej triedy a hyperrovinou je maximalizovaná.

Tieto najbližšie body sa nazývajú "podporné vektory", pretože práve ony určujú polohu a orientáciu rozhodovacej hyperroviny. Ak sa niektorý z týchto bodov posunie, zmení sa aj poloha rozhodovacej hranice. Preto sú tieto body tak dôležité pre model SVM, odtiaľ aj názov "podporné vektory".

SVM môže pracovať aj v nelineárnom režime prostredníctvom techniky nazývanej "kernel trick". Táto technika mapuje vstupné dáta do vyššieho dimenzionálneho priestoru, kde sú dáta lineárne separovateľné.

Výhodou SVM je, že poskytuje veľmi robustnú a efektívnu metódu pre klasifikáciu, najmä pre problémy s vysokými dimenziami a keď je rozdelenie dát komplexné. Okrem toho, keďže SVM sa sústredí iba na podporné vektory, nie je to tak citlivé na prítomnosť odľahlých hodnôt, ako niektoré iné metódy klasifikácie.

---

31. SVM – klasifikátor mäkkého okraja
32. SVM – stroje podporných vektorov (nelineárny a s kernelom)
33. Generovanie rozhodovacej procedúry vo forme etalónu
34. Dvojtriedna a multitriedna klasifikácia, spriemerňovanie príkladov
35. Neinkrementálna indukcia etalónov (algoritmus NCD)
36. Inkrementálna indukcia etalónov (algoritmus ICD)
37. Lenivé učenie ako extenzionálna reprezentácia a ich použitie
38. Algoritmus kNN (k najbližších susedov)
39. Metriky podobnosti
40. Regresná analýza (lineárna a logistická)
41. Učenie súborom metód – Stacking
42. Učenie súborom metód – Bagging

---

43. Učenie súborom metód – Boosting

Boosting je metóda zoskupenia používaná na zlepšenie presnosti **slabých** klasifikátorov. Myšlienka boostingu spočíva v postupnom trénovaní série slabých klasifikátorov, ako sú napríklad rozhodovacie stromy, pričom každý nasledujúci klasifikátor sa zameriava na **nesprávne** klasifikované príklady z **predchádzajúceho** klasifikátora.

---

44. Učenie súborom metód – Random Forests (Náhodné lesy)

Náhodný les je **súbor**, ktorý kombinuje viacero rozhodovacích stromov a **agreguje** ich výstupy s cieľom zlepšiť presnosť a robustnosť konečného modelu.

Náhodné lesy náhodne vyberajú podmnožinu **príznakov** a podmnožinu trénovacích **príkladov** na vytvorenie každého rozhodovacieho stromu.

Na vytvorenie predpovede pre nový príklad náhodný les agreguje výstupy **všetkých** rozhodovacích stromov.

---

45. Aktívne učenie

Aktívne učenie je technika, pri ktorej algoritmus aktívne vyberá príklady, ktoré chce **označiť**, namiesto toho, aby sa spoliehal na **vopred** označený súbor údajov.

Počas aktívneho učenia algoritmus iteratívne vyberá malú **podmnožinu** neoznačených príkladov, ktoré má označiť **expert**. Algoritmus potom použije tieto dodatočné označené údaje na aktualizáciu modelu a výber **ďalšej** sady príkladov na označenie.

---

46. Posilňované učenie (Reinforcement Learning)

- **Učenie odmenou a trestom**
- Rieši sekvenčnú úlohu
- Operuje v stavovom priestore
- Hľadá najkratšiu cestu od počiatočného k cieľovému stavu
- Viac počiatočných stavov a jeden cieľový
- Jeden počiatočný a viac cieľových stavov (šach)
- Učenie spočíva vo vylepšovaní rozhodnutí
- Rozhodnutia – v mozgu agenta (softvér), vo fyzickom svete

Použitie:

- Aplikuje sa presné rozhodnutie v každom stave podľa výsledku učenia

- Výsledok učenia je optimálna cesta ako séria rozhodnutí

- Vstupy: stavový priestor – mapuje doménu
  čiastočné znalosti problémovej domény
- Výstup: presné rozhodnutie v každom stave
- Agent počas hľadania generuje vlastné experimenty
  – interná odmena
- Môže aj sledovať riešenie doménového experta
  - externá odmena
- Riadiace znalosti môžu byť získané aj vyhodnotením
- úspešných ciest (výhry)
- neúspešných ciest (prehry, sľučky)
- Riešenie sekvenčnej úlohy vyžaduje viac krokov (rozhodnutí) v porovnaní s klasifikačnou úlohou
- Informáciu o úspešnosti kroku dostane riešiteľ dlho po jeho vykonaní
- Sústreďuje sa na preferenciu znalostí na výber operátora pomocou ohodnocovacej funkcie
- Stratégia - v každom kroku zvoliť stav s najvyššou odmenou
- Reprezentácia – pomocou tabuľky
- Popisuje páry stav ***s*** – akcia ***a***
- Každá bunka tabuľky obsahuje očakávanú odmenu, reprezentujúcu vhodnosť vykonania akcie ***a*** v danom stave ***s***
- Takúto tabuľku je možné zobraziť ako orientovaný ohodnotený graf, ktorého uzly znázorňujú stavy a hrany akcie
- Tento graf sa nazýva stavový priestor

---

47. Q-learning a „Bucket Brigade“

Q-learning a "Bucket Brigade" sú dva príklady algoritmov učenia s posilnením (reinforcement learning).

1. Q-learning: Q-learning je algoritmus učenia s posilnením, ktorý sa používa na riešenie úloh, kde sa agent snaží nájsť optimálnu stratégiu pre rozhodovanie v dynamickom prostredí. Agent sa učí, ako maximalizovať očakávané dlhodobé odmeny tým, že sa učí hodnotiť rôzne akcie v závislosti od stavu prostredia. Q-learning algoritmus sa snaží nájsť optimálnu stratégiu v podobe funkcie Q(s,a), ktorá popisuje očakávanú dlhodobú odmenu, ak agent vykoná akciu a vstúpi do stavu s.

2. "Bucket Brigade": "Bucket Brigade" je algoritmus učenia s posilnením, ktorý sa používa na riešenie problémov kooperatívneho riadenia. V "Bucket Brigade" algoritme sa skupina agentov snaží nájsť spoločnú stratégiu na dosiahnutie cieľa. Algoritmus používa spätnú väzbu, ktorá sa prenáša medzi agentmi, aby sa zlepšila kvalita stratégie. "Bucket Brigade" algoritmus sa spolieha na to, že každý agent bude schopný zlepšiť svoju vlastnú stratégiu, keď vidí, že iní agenti sú úspešní pri dosahovaní cieľa.

Tieto dva príklady algoritmov sú zamerané na učenie s posilnením, kde sa agent snaží maximalizovať odmenu v dynamickom prostredí. Q-learning sa používa na úlohy, kde agent musí robiť rozhodnutia na základe stavu prostredia, zatiaľ čo "Bucket Brigade" sa používa na problémy kooperatívneho riadenia, kde sa skupina agentov snaží dosiahnuť spoločný cieľ. Tieto algoritmy môžu byť veľmi užitočné pri riešení rôznych problémov, ktoré sa vyskytujú v dynamických prostrediach.

---

48. Definícia a typy zhlukovania
49. Zhlukovanie aglomeratívne, divízne, paralelné a sekvenčné
50. Iteratívne K-means
51. Divízne K-means
52. Aglomeratívne zhlukovanie
53. Hierarchické zhlukovanie - Cobweb
54. Pravdepodobnostné zhlukovanie

---

55. Výpočtová teória učenia

Krátky teoreticky popis:

Výpočtová teória učenia je oblasť, ktorá spojuje matematiku a informatiku. Zaoberá sa tým, ako stroje môžu byť trénované na riešenie úloh, ako napríklad rozpoznávanie obrazov, hlasu alebo odporúčanie produktov.

Analyza(Ako to funguje):

Výpočtová teória učenia sa zameriava na to, ako môžeme zistiť, aké problémy sú riešiteľné pomocou strojového učenia a aké sú ich teoretické hranice. Jedným z modelov používaných v tejto oblasti je PAC model, ktorý určuje, ako veľa trénovacích dát je potrebných na dosiahnutie určitej úrovne presnosti.

Výsledky:

Vysledky tejto oblasti sú dôležité pre návrh algoritmov strojového učenia a pre pochopenie ich vlastností. Teoretické hranice pomáhajú určiť, kedy je použitie strojového učenia efektívne a kedy nie.

Výpočtová teória učenia tiež umožňuje presne určiť, aké typy úloh sú riešiteľné pomocou strojového učenia a aké nie.

Výpočtová teória učenia sa rýchlo rozvíja vďaka vývoju nových algoritmov a technológií, a preto bude mať v budúcnosti zásadný vplyv na oblasť strojového učenia a umelú inteligenciu ako celok.

---

56. Definícia problému a PAC (**P**robable Learning of **A**pproximately **C**orrect Hypothesis)

PAC učenie (Probable Learning of Approximately Correct Hypothesis) sa týka strojového učenia a je to teória, ktorá sa zaoberá tým, ako strojové algoritmy môžu naučiť sa odhadovať správne riešenia na základe obmedzených a neúplných dát.

Jednou z kľúčových myšlienok PAC učenia je, že učenie sa môže byť úspešné aj v prípade, že niektoré údaje sú nesprávne alebo chýbajú, ak sa naše hypotézy približujú k správnemu riešeniu s určitou pravdepodobnosťou.

Problém PAC spočíva v tom, ako nájsť najlepšiu hypotézu z množiny hypotéz, ktoré sú aproximáciou riešenia. Cieľom je nájsť hypotézu, ktorá bude mať nízku chybu a bude sa dobre generalizovať pre nové dáta, ktoré neboli použité pri učení.

---

57. Odhad chyby hypotézy a hodnotenie efektívnosti

Hodnotenie účinnosti experimentu zahŕňa určenie, do akej miery bol úspešný pri dosahovaní svojich cieľov. Často sa to robí pomocou štatistických testov, ktoré porovnávajú výsledky experimentu s očakávanými výsledkami pri nulovej hypotéze.

Na druhej strane odhad chyby hypotézy sa týka toho, ako presne možno predpovedať výsledok experimentu a jeho interpretáciu. Na odhad chyby hypotézy sa používajú štatistické metódy, ako sú opakované merania alebo použitie kontrolných skupín. Je však dôležité poznamenať, že hodnotenie účinnosti by sa malo vždy interpretovať v kontexte odhadovanej chyby hypotézy.

V praxi sa tieto dve metódy často používajú v kombinácii s inými štatistickými metódami a technikami, ako je regresná analýza a analýza rozptylu, na určenie pravdepodobnosti, že pozorovaný účinok bol spôsobený konkrétnou premennou.

Výsledky týchto štatistických analýz sa môžu prezentovať pomocou grafov, tabuliek alebo iných vizuálnych techník.

---

58. Komplexnosť príkladov v konečnom priestore hypotéz

---

59. Základné princípy kognitívnych algoritmov

P1 - usporiadanie priestoru pojmov - prehľadávanie priestoru všetkých kandidátov pojmov

- priestor pojmov je usporiadaný (napr. podľa všeobecnosti)
- nutnosť definovať operátory pre pohyb v priestore pojmov (napr. operátor špecifikácie/zovšeobecnenia)
- prehľadávanie od všeobecného k špecifickému alebo naopak

P2 - horolezecký princíp - založený na optimalizácii

- prehľadávacia stratégia založená na gradientovom hľadaní extrému v lokálnom okolí riešenia

P3 - delenie priestoru príkladov na podpriestory - priestor príkladov sa rekurzívne delí na podpriestory, kým nie je splnená ukončovacia podmienka (napr. minimálna entrópia)

P4 - riadenie výnimkami - pre chybne klasifikované príklady sa vytvoria nové (pseudo) triedy
P5 - súťaživý princíp - kandidáti pojmov sa ohodnotia pomocou zvolenej hodnotiacej funkcie a vyberie sa najlepšie ohodnotený pojem
P6 - skórovacia funkcia - umožňuje vytvárať systémy s prehľadávacíme preferenciami, ktorý bude pojmy ohodnotené skórovacou funkciou uvažovať skôr
P7 - redukcia počtu kandidátov - v každej iterácii algoritmu sa obmedzí počet pojmov na určitý počet

---

60. Usporiadanie priestoru pojmov

    Usporiadanie pojmov v priestore sa uskutočňuje pomocou operátorov **zovšeobecnenia**, resp. operátorov **špecifikácie**. Často citovaná dizertačná práca (Winston, 1970) uvádza prehľadávanie použitím operátorov zovšeobecnenia a špecifikácie. Zovšeobecnenie a špecifikácia sú najbežnejšie typy operácií na pohyb v priestore pojmov.

---

61. Horolezecký princíp

---

62. Delenie priestoru príkladov na pod-priestory (rozdeľuj a panuj)

    Delenie priestoru príkladov na podpriestory je bežný prístup v strojovom učení a je často používaný v kontexte "rozdeľuj a panuj" stratégií. Tento prístup spočíva v rozdelení pôvodného problému na menšie, jednoduchšie riešiteľné problémy, a potom v spojení týchto riešení do konečného výsledku.

    Jedným z najznámejších príkladov tohto prístupu je strom rozhodnutí. Strom rozhodnutí delí priestor príkladov na podpriestory na základe hodnôt rôznych atribútov. Každý uzol v strome rozhodnutí predstavuje otázku týkajúcu sa hodnoty jedného z atribútov, a každá vetva z tohto uzlu predstavuje možnú odpoveď. Týmto spôsobom, strom rozhodnutí delí priestor príkladov na menšie podpriestory, ktoré sú "čistejšie" v tom zmysle, že obsahujú príklady, ktoré sú viac podobné vzhľadom na cieľový atribút.

    Podobný prístup je používaný v metóde k-najbližších susedov (k-NN), kde priestor príkladov je implicitne rozdelený na oblasti založené na najbližších susedoch každého príkladu.

    Metódy založené na klastrovaní, ako je K-stredov, tiež delia priestor príkladov na podpriestory, kde každý podpriestor predstavuje skupinu príkladov, ktoré sú si navzájom podobné.

    V kontexte SVM, "rozdeľuj a panuj" môže byť použitý v metóde jedna-proti-všetkým (one-versus-all) alebo jedna-proti-jednej (one-versus-one), kde viacklasový klasifikačný problém je rozdelený na viacero binárnych klasifikačných problémov.

---

63. Riadenie výnimkami

---

64. Súťaživý princíp

- Kandidáti pojmov sa ohodnotia pomocou zvolenej hodnotiacej funkcie a vyberie sa najlepšie ohodnotený pojem.
- Príklad hodnotiacej funkcie
  – pravdepodobnosť triedy podmienená hodnotami atribútov klasifikovaného príkladu.
  – vzdialenosť (Euklidova) klasifikovaného príkladu od typických reprezentantov jednotlivých tried.
- Princíp využívajú algoritmy:
  Naivný Bayes klasifikátor (pravdepodobnostný pojem)
  NCD, ICD (etalóny)

---

65. Skórovacia funkcia a redukcia počtu atribútov
66. Voľba princípov a návrh algoritmu

---

67. Aplikačné možnosti strojového učenia
    - Klasifikácia podozrivej bankovej operácie - veľké množstvo trénovacích údajov z niekoľkých bánk: P1 & P6 & P7.
    - Diagnostika zriedkavých diagnóz špecializovanej oblasti medicíny – málo početná trénovacia množina a veľký rozptyl príkladov jednej triedy medzi príklady ostatných tried: P3 a v jednotlivých podpriestoroch P1 alebo P4
    - P4 formuje pre výnimky pseudo-triedu reprezentujúcu novú chorobu spôsobenú zmutovaným vírusom.

P1 - Usporiadanie priestoru pojmov:

- Prehľadávanie priestoru všetkých kandidátov pojmov.
- Priestor pojmov je usporiadaný (napr. podľa všeobecnosti).
- Nutnosť definovať operátory pre pohyb v priestore pojmov (napr. operátorov špecifikácie/zovšeobecnenia).
- Prehľadávame od všeobecného k špecifickému (G-S), od špecifického k všeobecnému (S-G), resp. obidvoma smermi.
- Princíp využívajú algoritmy: VSS, EGS a ESG.

P3 - Delenie priestoru príkladov na pod-priestory:

- Priestor príkladov sa rekurzívne delí na pod priestory, kým nie je splnená ukončovacia podmienka (napr. v každom pod priestore sú iba príklady jednej triedy).

- Podmienka delenia (testovací atribút) sa vyberá pomocou informačnej teórie (napr. minimálna entrópia).
- Princíp využívajú algoritmy:

    NSC, AQ11 (alternatívne klasifikačné pravidlá)

    ID3, ID5R a C4.5 (rozhodovacie stromy)

P4 - Princíp učenia riadeného výnimkami:

- Pre chybne klasifikované príklady - výnimky sa vytvoria nové triedy (pseudo-triedy).
- Tento proces sa opakuje, kým nie sú všetky príklady správne klasifikované (nové iterácie neprinášajú lepšie výsledky, maximálny počet iterácií).
- Princíp využívajú algoritmy:

    NCD, ICD (etalóny)

    NEX (rozhodovacie zoznamy)

P6 - Skórovacia funkcia:

- Umožňuje vytvárať systémy s prehľadávacími preferenciami (search bias), ktorý bude pojmy lepšie ohodnotené skórovacou funkciou uvažovať skôr.
- Mäkké preferencie (Soft Bias) – pojmy s vyšším skóre majú prednosť.
- Vo všeobecnosti je skóre priamo úmerne závislé na počte pokrytých pozitívnych príkladov a nepriamo úmerne závislé na počte pokrytých negatívnych príkladov.
- Zložitejšie prístupy používajú štatistické alebo informačné miery (entrópia, signifikancia).
- Princíp využívajú algoritmy:

HGS, HSG (klasifikačné pravidlá)

HCT (tabuľka kritérií)

ID3, ID5R, C4.5 (rozhodovacie stromy)

CN2 (rozhodovacie zoznamy)

P7 - Redukcia počtu kandidátov:

- V každej iterácii algoritmu sa obmedzí počet pojmov na určitý počet (Beam Size – BS).

- Tvrdé preferencie (Hard Bias) – niektoré typy pojmov sú vopred vylúčené z prehľadávania.
- Z kandidátov pojmov usporiadaných podľa hodnôt skórovacej funkcie sa vyberie iba BS najsľubnejších pojmov.
- Horolezecký princíp je špeciálnym prípadom pri BS=1.
- Princíp využívajú algoritmy:

HGS, HSG (klasifikačné pravidlá)

HCT (tabuľka kritérií)