Autoři: RNDr. Eva Janoušová, Mgr. et Mgr. Petr Dluhoš
Institut biostatistiky a analýz, Masarykova univerzita, Brno

Algoritmus podpůrných vektorů (SVM) – příklad

Bylo provedeno měření objemu hipokampu a amygdaly (v cm^3) u 3 pacientů s Alzheimerovou chorobou ( ) a 3 kontrolních subjektů ( ) (označení D – diseased, H – healthy). Naměřené hodnoty objemu hipokampu a amygdaly u pacientů (resp.) a kontrol (resp.) byly zaznamenány do matic resp. :

Určete, zda testovací subjekt patří do skupiny pacientů či kontrolních subjektů pomocí algoritmu podpůrných vektorů.

Řešení:

Algoritmus podpůrných vektorů (support vector machine – SVM) se snaží najít mezi všemi možnými dělícími nadrovinami (ve dvourozměrném případě přímkami) takovou, která rozděluje subjekty do dvou požadovaných tříd co nejrobustněji - tj. takovou, která prochází v co největší vzdálenosti od subjektů z obou tříd. V případě, že přímka dělící subjekty do požadovaných tříd existuje (třídy jsou lineárně separovatelné), postupuje se takto:

Orientaci dělící přímky si označíme vektorem a její polohu číslem . (Klasifikace subjektu do třídy , resp. bude dána tím, jestli je výraz větší, resp. menší, než 0.) Podle vztahu 2.138 na straně 43 ve skriptech je vzdálenost bodu od dělící roviny dána vztahem ———, kde je velikost vektoru . Změna velikosti tohoto vektoru nijak neovlivní výslednou klasifikaci a tak si ji můžeme stanovit libovolně, například tak, aby pro nejbližší bod ze třídy byla hodnota výrazu rovna a pro nejbližší bod ze třídy byla hodnota výrazu rovna . V tom případě máme na každé straně od dělící přímky toleranční pásmo o šířce ——, ve kterém se nenachází žádný bod. Pro všechny body z trénovací množiny platí:

což můžeme stručněji zapsat jako

kde = 1 pro ze třídy ( ) a = -1 pro ze třídy

Abychom dosáhli co nejrobustnější klasifikace, budeme hledat takové hodnoty a , aby byla celková šířka tolerančního pásma — — — co největší. Hledat maximum funkce — je to stejné, jako hledat minimum funkce — a toto minimum se nezmění, když kladnou hodnotu v čitateli umocníme na druhou (což nám zjednoduší výpočty). Takže dostáváme kriteriální funkci:

—

jejíž hodnotu se snažíme minimalizovat. Zároveň ale musí pro všechny body z trénovací množiny platit výše popsané podmínky:

Tuto podmíněnou kvadratickou optimalizační úlohu můžeme řešit metodou Lagrangeova součinitele: Zavedeme vektor Lagrangeových součinitelů a pomocí nich vyjádříme optimalizovanou funkci jako:

—

za podmínek .

Toto Lagrangeovu funkci zderivujeme podle proměnných a a derivace položíme rovny nule, čímž získáme soustavu:

—

—

Po zderivování získáme:

což je 9 (nelineárních) rovnic o 9 neznámých.

Poznámka: Povšimněme si, že pro výpočet orientace dělící přímky jsou důležité jen ty body, pro které platí . Každý takový bod ovšem musí podle příslušné rovnice výše splňovat podmínku , tedy musí ležet přesně na hranici tolerančního pásma. Takovým bodům říkáme podpůrné vektory a jen na nich závisí umístění dělící přímky.

Pro řešení soustavy bychom mohli použít nějaký software pro řešení obecných úloh kvadratického programování (QP solver), ale v praxi se většinou úloha převádí na duální problém, který lze řešit snadněji a jehož řešení se shoduje s řešením původní úlohy:

Dosadíme-li výrazy a do Lagrangeovy funkce, získáme

–

–

–

což je funkce závisející pouze na Lagrangeových součinitelích Duálním optimalizačním problémem k původnímu problému je najít hodnoty maximalizující za podmínek:

Poznámka: vektory se ve funkci vyskytují pouze ve formě skalárního součinu, který může být nahrazen nějakou nelineární (jádrovou) funkcí, čímž docílíme nelineární separace tříd.

Pro řešení úlohy jsou potřeba vzájemné skalární součiny vektorů z testovací množiny. Spočítáme je a zapíšeme do tzv. Gramovy matice:

Ovšem skalární součiny se ve funkci vyskytují vždy v součinu s . Tuto informaci zahrneme do Gramovy matice:

kde značí násobení po prvcích. Dosadíme do funkce a úlohu vyjádříme v maticové podobě:

a .

Kvadratickou optimalizační úlohu v tomto tvaru lze řešit opět pomocí obecného programu (v Matlabu funkce *quadprog*) nebo pomocí nějaké specializované metody pro úlohy v tomto tvaru - například SMO – Sequantial minimal optimization (v matabu součástí funkce pro trénování SVM *svmtrain*).

Použijeme funkci *quadprog*, do které dosadíme příslušné hodnoty (s opačným znaménkem, protože funkce hledá minimum):

kde popisují minimalizovanou funkci, značí, že nemáme žádná omezení ve tvaru nerovnic (zde se neberou v úvahu jednoduchá omezení jednotlivých proměnných), popisují soustavu omezení ve tvaru rovnic a poslední vektor popisuje jednoduchá omezení jednotlivých proměnných .

Výsledkem výpočtu funkce *quadprog* jsou hodnoty – — , pro které nabývá funkce při dodržení omezení svého maxima. Podpůrnými vektory jsou tedy body , protože jim příslušející jsou nenulové. Vypočítáme orientaci dělící přímky:

– — – —

Polohu dělící přímky určíme ze znalosti toho, že pro libovolný podpůrný vektor musí platit tedy například pro :

Nyní můžeme klasifikovat subjekt :

Protože testovací subjekt bude zařazen do třídy kontrolních subjektů.

Ověříme, že natrénovaný klasifikátor správně zařadí všechny subjekty z trénovací množiny:

Všechny subjekty jsou zařazeny správně. Povšimněme si, že hodnot nabývají právě ty subjekty, které tvoří podpůrné vektory.