# Poročilo - DN2 - Izbrane teme iz analize podatkov

Matej Novoselec

6. maj 2024

Koda za izpis izračunov in izris grafov je na voljo v datoteki *dn2.ipyb*. Pogon kode traja $< 160s$.

## Naloga 1: SVM

### 1.a)

Dovolimo si najprej odgovoriti na prvo vprašanje iz naloge 1.b), saj nam bo odgovor pomagal pri razvrstitvi.

Kot omenjeno na predavanjih, vrednost parametra C vpliva/določa kompromis med pravilnostjo klasifikacije podatkov in velikostjo mere do katere sta razreda podatkov ločena. Majhna vrednost parametra C se osredotoča/preferira večjo razliko med razredi podatkov, četudi to pomeni, da so posamezni podatki napačno klasificirani. Sledi, da lahko takrat na grafu/sliki odločitvene meje modela pogosto opazimo več podpornih vektorjev. Obratno večja vrednost parametra C preferira pravilnost klasifikacije in na ta način ne poskrbi nujno, da je mera razlike med razredi podatkov največja. Na grafu/sliki odločitvene meje takega modela pogosto pričakujemo manj podpornih vektorjev.

Sedaj sliki vsake odločitvene meje priredimo jedro, s katerim je bila naučena. Ravna črta na slikah (a) in (e) očitno nakazuje, da gre za linearno odvisnost. Sklepamo, da sta pripadajoči jedri linearni. Dodatno opazimo, da je pri odločitveni meji na sliki (e) opaziti veliko več podpornih vektorjev kot na sliki (a). Zgoraj zapisano nam da podlago, da zaključimo (a) $\leftrightarrow$ (2) in (e) $\leftrightarrow$ (1).

Sedaj se posvetimo slikam (b), (f) in (d). Oblike vseh odločitvenih mej odražajo obliko grafa polinomske funkcije - sliki (b) in (f) odražata obliko parabole, slika (d) pa ima obliko hiperbole. Na prvi pogled bi zaradi podobnosti odločitvenih mej (b) in (f) radi trdili, da gre pri obeh za polinomsko jedro stopnje 2, vendar za ta zaključek navedimo dodatni argument.

Dodatna stopnja polinomskega jedra nam omogoča boljše prileganje podatkom, ki pa je velikrat razvidno iz večje nazobčanosti odločitvene krivulje. Ta navadno na pogled deluje še dodatno nepravilno/nelinearno. Nižjo stopnjo polinoma povezujemo z odločitveno mejo, ki odraža bolj gladko krivuljo, kot bi denimo to pripisali odločitvenim mejam na slikah (b) in (f). Obe značilnosti, ki ju pripisujemo jedrom polinoma višje stopnje, sta razvidni iz slike (d) - zavoj odločitvene meje je vidno bolj oster, a se boljše prilega podatkom kot denimo

to storita odločitveni meji na slikah (b) in (f). Sklepamo torej (d) $\leftrightarrow$ (5), s štetjem odločitvenih vektorjev na slikah (b) in (f), pa z zgornjim sklepom (prek vrednosti parametra C), zaključimo (b) $\leftrightarrow$ (4) in (f) $\leftrightarrow$ (3).

Odločitveno mejo iz slike (c) smo iz zgornje analize izločili zaradi opazno visoke stopnje nelinearnosti, ter nesimetričnosti. Če bi slika (c) odražala polinomsko jedro stopnje 2, potem bi pričakovali, da bo krožna oblika zajema podatkov odražala elipso ali krožnico, vendar je že na pogled jasno, da temu ni tako. Dodatno so zavoji na sliki ostrejši, ter dodatno odražajo nepravilnost. Zaključimo torej (c) $\leftrightarrow$ (6).

### 1.b)

Naloga nas sprašuje po prednostih odločitvene meje z večjo vrednostjo parametra C, ter prednostih odločitvene meje z manjšo vrednostjo parametra C.

V duhu zgoraj navedenega - manjše vrednosti parametra C skrbijo/preferirajo večjo mero razlike med razredi podatkov (zato hitreje prihaja do napačne klasifikacije podatkov), kar lahko pripelje do boljše posplošitve na nove podatke; večja vrednost parametra C preferira natančnejšo klasifikacijo podatkov (ter s tem ne določi nujno maksimalnega razmika med razredi podatkov), s čemer lahko zaznamo kompleksnejše vzorce v podatkih, a s tem tvegamo preprileganje, ter več napak na novih podatkih.

## Naloga 2: Ansambli

### 2.c)

Z uporabo razreda iz 2.b) in funkcij iz 2.a) definiramo funkcijo *celotno_vrecenje*, ki bo za podan osnovni model, ter število spremenljivk in modelov, skonstruirala, uporabila in testirala naš ansambelski model. Za testiranje/učenje uporabimo KFold, ter model vrednotimo prek RMSE. Ker se tako v učnih, kot tudi testnih podatkih pojavijo vrednosti Nan, le-te ob pojavitvi zamenjamo z 0. Omenimo še, da smo za ponovljivost poskusov nastavili *random seed* tako za knjižnico *random*, kot tudi *numpy*.

Sedaj najprej testiramo, pri katerem številu spremenljivk, je povprečen RMSE modela najnižji (ker je vseh spremenljivk 8, si ogledamo vrednosti $1, \ldots, 8$). Tu je vredno omeniti, da moramo na roke nastaviti število modelov, pri katerem bomo spremljali ansambelske modele z različnim številom spremenljivk. Izbor te vrednosti vpliva na vrednost, v kateri glede na število spremenljivk dobimo najmanjšo napako (ta se s spremembo števila modelov navadno zamakne za 1 v levo ali desno). V kodi smo vzeli 25 modelov, ter grafično predstavili in izpisali odvisnost vrednosti RMSE, od števila naključno izbranih napovednih spremenljivk. Opazimo, da minimalno vrednost RMSE, ki znaša 3.85, dobimo, ko naključno izberemo 4 spremenljivke.

Sedaj fiksirajmo število spremenljivk (vzemimo kar 4, ki je minimum iz prejšnega vprašanja), ter si oglejmo, pri katerem številu modelov dobimo povprečno

najnižji RMSE. Da bodo razlike čim bolj opazne, za število modelov vzamemo dokaj velik razpon vrednosti. Po izpisu vrednosti in izrisu grafa opazimo, da minimum, z vrednostjo 3.717, zavzamemo ko vzamemo 90 modelov. Iz narisanega grafa je tudi jasno razvidno, da vrednost RMSE z začetnim večanjem števila modelov zelo hitro (lahko bi dejali eksponentno) pada. Okvirno pa bi lahko rekli, da se napaka ne spreminja več bistveno ko dosežemo 100 modelov, oziroma se še popolnoma umiri, ko dosežemo 500 modelov.

Kot to narekuje navodilo, si ogledamo povprečen RMSE pri modelu le enega drevesa, ter ga primerjamo z RMSE ansambelskega modela. Opazimo, da je RMSE pri ansambelskemu modelu veliko manjši (torej 3.717), pri modelu enega odločitvenega drevesa pa znaša kar 5.13.

### 2.d)

Razred iz naloge 2.c) nekoliko spremenimo in preimenujemo v *UtezenoVrecenje*. Ideja je na smiselen način obtežiti vsakega izmed modelov ansambelskega modela, ter tako upati na izboljšavo celotnega modela.

Uteži bomo določili sorazmerno z RMSE, ki jo vsak izmed napovednih modelov doseže na testni množici. Nato bomo za napoved/ansambelski model vzeli kar (s tako konstruiranimi utežmi) obteženo povprečje posameznih modelov.

Ponovno najprej najdemo število spremenljivk, pri kateri je RMSE takega ansambelskega modela najmanjši, nato pa najdemo še optimalno število modelov za dano število spremenljivk. Po izrisu grafov in izpisu posameznih napak opazimo, da povprečen RMSE tako izbranega obteženega ansambelskega modela znaša le še 3.378. Obtežitev ansambelskega modela tako predstavlja izboljšavo še za okoli 0.5.

## Naloga 3: Nevronske mreže

### 3.b)

Za začetek zapišimo nekaj osnovnih opažanj iz podatkov. Podatki napovednih spremenljivk so urejeni v matrike, ki ob izrisu prikažejo človeške obraze. Slike so velike $48 \times 48$ pikslov, vsaki pa je pripisana ciljna spremenljivka, ki zavzame vrednosti $0, 1, \ldots, 6$. Smatramo, da gre za pripis čustev (ki so torej zamaskirana s števili od 1 do 6) vsaki izmed slik.

Naloga nas sprašuje po konstrukciji modela, s pomočjo nevronskih mrež, ki bo imel čim manjši *F1 score*, zato si pred začetkom oglejmo še koliko podatkov pripada vsakemu izmed razredov od 0 do 6. Z izrisom histograma opazimo, da je podatkov razreda 1 dokaj malo, ter podatkov razreda 3 nekoliko več od preostalih razredov. Zaključimo, da se *F1 score*, ki bo obenem upošteval še porazdelitev po razredih, ne bo popolnoma skladal s samo natančnostjo modela. Vendar je ta kljub temu še najbolj intuitivna mera uspešnosti modela, zato bomo spremljali in komentirali vrednosti obeh.

Sedaj se lotimo konstrukcije same nevronske mreže. Ker gre za strojno učenje na slikah, bom uporabili konvolucijsko nevronsko mrežo oziroma CNN. Ta je zapisana/definirana v istoimenskem razredu. Kot je hitro opaziti, je le-ta sestavljena iz kar velikega števila plasti, z različnim številom parametrov. Da preprečimo preprileganje podatkov, smo v mrežo vključili tudi *nn.Dropout* in *nn.BatchNorm2d*, ter za redukcijo prostorske dimenzije (kot je to navadno koristno pri CNN) vključili tudi *MaxPool2d*. Eksperimentiral/prilagajal sem število plasti, število parametrov na vsaki plasti, ter zaporedje zapisanih dodatnih operacij, a se je slednja dokazala za najuspešnejšo. Nevronske mreže sem med sabo primerjal najprej po natančnosti (ki vsaj načeloma okvirno odraža *F1 score*), nato pa sem prešel na vrednotenje modelov prek F1 score-a.

Iskanje (do sedaj) najboljšega modela je potekalo s postopnim učenjem glede na različne vrednosti spremenljivke *learning rate* v funkciji *train*, funkcija *test* pa je vsakič preverila, ali je na novo konstruirani model boljši od že obstoječega. Tega smo skozi proces hranili (in po potrebi na novo shranjevali) v datoteki *model.pth*. Modelu smo ob učenju spreminjali tudi vrednost parametra *batch_size*, ter poskusili tudi s predhodno transformacijo podatkov (ki je sedaj zakomentirana pod nalogo 3.a)). Ta je ob vsaki izbiri podatkov naključno rotirala, obrnila, zamaknila ali nekoliko pobližala sliko, kar naj bi pri učenju modela pripomoglo k raznolikosti in še dodatno preprečilo preprileganje.

Z opisanimi postopki smo skonstruirali model, ki na v naprej pripravljeni testni množici doseže natančnost $59,5\%$, ter obenem doseže *F1 score* $0,594$. Tu le omenimo, da smo za *F1 score* nastavili parameter *weighted*, ki *F1 score* izračuna na vsakem razredu posebej, ter nato naredu obteženo (glede na število vzorcev v vsakem razredu) povprečje le-teh. Kot to od nas zahteva naloga, izpišemo še *F1 score* za vsakega izmed razredov. Opazimo, da se posamezne vrednosti presenetljivo veliko razlikujejo, najvišji *F1 score* dosežemo na razredu 3, kar $0,797$, najnižjega pa na razredu 2, le $0,407$.

Zaključimo s komentarjem, kako bi se model verjetno dalo še izboljšati. Gotovo bi bilo vredno poskusiti z vnaprej naučenimi modeli, menim pa, da bi že samo daljše časovno obdobje učenja (skupaj s fleksibilnostjo opisanega transformatorja) pripeljalo do boljših rezultatov.