Poročilo - DN1 - Izbrane teme iz analize podatkov

Matej Novoselec

28. marec 2024

Koda za izpis izračunov in izris grafov je na voljo v datoteki *dn1.ipyb*. Pogon kode traja $< 200s$.

Naloga 1: Linearna regresija

1.a)

Spomnimo se teoretičnega ozadja iz predavanj. Opravka imamo z numeričnimi napovednimi spremenljivkami, ter iščemo optimalne vrednosti koeficientov modela linearne regresije. Rečeno nekoliko drugače, iščemo vrednosti koeficientov, da bo napaka napovednega modela najmanjša po metodi najmanjših kvadratov.

Na predavanjih smo izpeljali, da se to zgodi natanko tedaj, ko vrednosti koeficientov ("zložene" v matriko β) dobimo prek formule $\beta = \left(X^T X\right)^{-1} X^T Y$, kjer X predstavlja v matriko "zložene" vrednosti napovedne spremenljivke, ter Y predstavlja v matriko "zložene" vrednosti ciljne spremenljivke.

Za izračun zapisano formulo uporabi funkcija *beta*, dejanske vrednosti koeficientov, za naš primer, pa izračuna in vrne funkcija *najdi_koeficiente*.

Ob ogledu kode za konstrukcijo ciljne spremenljivke opazimo, da je izračunana kot linearna kombinacija napovednih spremenljivk. Pričakujemo torej, da bodo vrednosti koeficientov modela linearne regresije, pridobljeni iz *najdi_koeficiente*, blizu vrednostim koeficientov/skalarjev linearne kombinacije napovednih spremenljivk, ki določa ciljno spremenljivko. Pričakovano potrdimo z izpisom koeficientov. Razliko med pričakovanimi vrednostmi koeficientov in dejanskim izračunom pripišemo računski napaki.

1.b)

Podobno se lotimo drugega dela naloge. Ponovno opazimo, da je ciljna spremenljivka dobljena kot linearna kombinacija (tokrat z dodatnim konstantnim členom) napovednih spremenljivk. Slednje skupaj z navodilom naloge nakazuje, da je med napovedne spremenljivke vredno dodati vektor enic, ki bo prek enakega postopka kot zgoraj, omogočal izračun vrednosti dodatnega koeficienta modela linearne regresije. Slednje (prek funkcije *beta*) naredi funkcija

najdi_koeficiente2. Podobno kot zgoraj pričakujemo, da bodo izračunani koeficienti po vrednostih zelo blizu dejanskim koeficientom/skalarjem iz linearne kombinacije ki konstruira ciljno spremenljivko.

Tudi tokrat zapisano potrdi izpis koeficientov, ter izračun napake RMSE modela linearne regresije. Ta je ustrezno manjša od zahtevane vrednosti 10^{-10}.

1.c,d)

Pri d) delu naloge uporabimo funkcije iz naloge b). Uporabili bomo torej model linearne regresije z dodanim koeficientom oziroma konstantnim členom. Za izračun vrednosti koeficientov in napoved ciljne spremenljivke poskrbita funkciji *najdi_koeficiente2* in *napovej*, zato se posvetimo drugemu delu navodila. Model želimo naučiti na vseh podatkih, uvoženih pri c) delu, obenem pa vemo, da bodo novi podatki prihajali iz iste verjetnostne porazdelitve kot učni podatki. Da bomo za "učenje" uporabili vse podatke, bomo uporabili prečno preverjanje, podatek o porazdelitvi pa nakazuje, da je smiselno točnost (ali napako) modela oceniti s povprečno vrednostjo napak posameznega "odseka". Ob ogledu izpisa v datoteki opazimo, da ocena napake znaša $0,2407$, tudi varianca napak pa je dokaj nizka.

Opomba: poskusil sem tudi s stratificiranim vzorčenjem - težava je, da so vrednosti ciljne spremenljivke numerične (python vrne da zvezno porazdeljene), zato je potrebna predpriprava podatkov in razdelitev v skupine. Ko sem to storil (z zaokroževanjem vrednosti) se izkaže da je ocena napake podobna, varianca pa sicer manjša, vendar ne bistveno (ker je tako ali tako že prvotna nizka). Glede nato da so podatki pred razdelitvijo v odseke premešani, ter jih je kar veliko, bi tako ali tako lahko trdili, da je razdelitev po odsekih za *KFold* skladna z verjetnostno porazdelitvijo podatkov, oziroma igra vlogo *StratifiedKFold*.

Naloga 2: Logistična regresija

2.a)

Izpisane so povprečne vrednosti, ter variance posameznih napovednih spremenljivk. V širšem kontekstu je slednje še lažje razbrati iz grafa, na katerem je za vsako napovedno spremenljivko narisana podatkom pripadajoča škatla z brki. Povprečna vrednost je največja pri tretji napovedni spremenljivki, podatki pa so daleč najbolj razpršeni pri drugi napovedni spremenljivki. Porazdelitve napovednih spremenljivko se opazno razlikujejo, nekaj podobnosti lahko opazimo le pri porazdelitvi zadnjih treh napovednih spremenljivk.

Ker je ciljna spremenljivka binarna, le izpišimo število pozitivnih in negativnih pojavitev, ter izračunajmo razmerje med pozitivnimi in negativnimi primeri. Opazimo, da je veliko (več kot štirikrat) več negativnih primerov.

2.b)

Problema se lotimo s stratificiranim vzorčenjem. Da bodo eksperimenti ponovljivi uporabimo *random.seed(18)*, da bo stabilnost modela logistične regresije čim večja (torej varianca napak med odseki čim manjša), pa uporabimo stratificirano vzorčenje. Za ovrednotenje točnosti modela uporabimo povprečje točnosti na vsakem izmed odsekov (*accuracy_score*). Naloga narekuje, da naj število vzorcev ne presega 10, zato si oglejmo točnosti in stabilnosti modela za število odsekov/vzorcev med 5 in 10. Izpišimo povprečne točnosti in variance točnosti za različna števila odsekov. Opazimo, da so variance napak (pričakovano) majhne, povprečne vrednosti točnosti pa se gibajo okoli 0.96. Povprečna točnost je največja pri petih odsekih in znaša 0.9616. Zapisana situacija je razvidna tudi iz grafa.

2.c)

Za začetek uporabimo namig in med napovedne spremenljivke (na konec) dodajmo še kvadrat vrednosti prve napovedne spremenljivke in produkt prvih treh napovednih spremenljivk. Sedaj postopamo enako kot zgoraj - uporabimo stratificirano vzorčenje na številu odsekov od 5 do 10. Za vsak število odsekov izpišemo povprečno točnost in varianco točnosti.

Hitro opazimo, da je povprečna vrednost točnosti opazno večja od tiste v prejšni podnalogi, varianca pa sedaj skoraj ničelna. Povprečna točnost modela se tokrat giblje okoli 0.996, največja pa je pri osmih odsekih - znaša kar 0.997.

Ovrednotimo še izboljšanje modela (po dodaji "nelinearnih" napovednih spremenljivk) oziroma izračunajmo razliko povprečne točnosti (za vsako število odsekov). Razlika se giblje med 0.032 in 0.037. Izboljšanje je jasno razvidno iz narisanega grafa.

2.d)

Analize pomembnosti koeficientov modela logistične regresije iz naloge 2.c) se lahko lotimo na dva načina. Po eni strani se lahko vprašamo, kateri izmed koeficientov imajo po absolutni vrednosti največjo vrednost, ter na ta način sklepamo, katere izmed napovednih spremenljivk so pomembne (in katere niso - postavimo mejo glede na absolutno vrednost koeficienta) za klasifikacijo. Po drugi strani, pa lahko vzamemo nekoliko bolj statističen pristop, ter si za posamezno napovedno spremenljivko/njen koeficient v modelu, ogledamo pripadajočo p-vrednost. Ta nam za koeficient pove, kako verjetno je, da je njegova vrednost takšna kot smo jo izračunali, ob hipotezi, da je v teoriji koeficient pripadajoče napovedne spremenljivke enak 0. Torej, nižja kot je p-vrednost pripadajočega koeficienta napovedne spremenljivke, bolj gotovo lahko trdimo, da je napovedna spremenljvka za klasifikacijo pomemba. Oziroma obratno, višja kot je p-vrednost koeficienta napovedne spremenljivke, bolj gotovo je, da smo do te vrednosti koeficienta prišli slučajno, ter da v resnici napovedna spremenljivka

za klasifikacijo ni pomembna. Ponovno je potrebno izbrati mejo, ki bo ločila pomembne napovedne spremenljivke od nepomembnim, navadno izberemo $0,05$.

Zgoraj zapisano sedaj poračunajmo (napovedne spremenljivke so označene z zaporedno številko v tabeli, za zadnji dve pa je zapisana formula). Opazimo, da je absolutna vrednost koeficienta pri napovedni spremenljivki $X4$ daleč najvišja, tudi pripadajoča p-vrednost koeficienta pa majhna. Sklepamo, da je slednja najbolj pomembna za klasifikacijo. Najmanjša absolutna vrednost koeficienta pripada napovedni spremenljivki $X3$, največja p-vrednost pa pripada napovedni spremenljivki $X6$. Opazimo, da kar za tri napovedne spremenljivke ($X3$, $X5$ in $X6$) p-vrednost presega vrednost $0,05$. Dodajmo še, da izpis absolutnih vrednosti koeficientov nakazujejo, da napovednim spremenljivkam, z največjimi p-vrednostmi, ne pripadajo nujno najmanjše absolutne vrednosti koeficientov.

Sedaj poskusimo dvoje. Najprej za pomembne napovedne spremenljivke pri klasifikaciji razglasimo le tiste s p-vrednostjo manjšo od $0,05$, nato pa še tiste, katerih absolutna vrednost koeficientov je večja od 0.3.

Po izpisu napak za različno število odsekov (in izrisu grafa) je jasno, da sta obe izbrani "maski"/izbiri pomembnih in nepomembnih napovednih spremenljivk za klasifikacijo v večini pozitivno prispevali k povprečni točnosti modela. Največjo vrednost ta doseže pri 5 odsekih, z masko glede na p-vrednosti. Ta znaša kar $0,997$. Ocena točnosti tako izbranega/nastalega modela, ter modela brez maske se ne razlikuje bistveno (za manj kot 0.005), kar lahko pripišemo dejstvu, da je bila točnost že prvotnega modela zelo blizu optimalni (blizu 1).

Naloga 3: K-najbližjih sosedov

3.a,b)

Ko uvozimo podatke opazimo, da so podatki v resnici (po blokih) urejeni po ceni. Zato je iz napovednih spremenljivk pošteno/smiselno odstraniti zaporedno številko diamanta v bazi (drugače na ceno v veliki meri vpliva le ta napovedna spremenljivka - s spodaj opisanim postopkom sem na ta način RMSE spravil na 50). Za začetek sedaj izrišimo histograme vsake izmed napovednih spremenljivk, da dobimo občutek za podatke.

Pred začetkom napovedovanja cene diamanta, se posvetimo še predprocesiranju podatkov. Ker so podatki napovednih spremenljivk različno porazdeljeni, se potencialnim težavam izognemo s standardizacijo podatkov (tu sem vse podatke - kategorične in numerične - standardiziral na enak način; poskusil sem storiti tudi ločeno, vendar slednje ni pozitivno vplivalo na oceno RMSE).

Druga težava leži v dejstvu, da so napovedne spremenljivke lahko med seboj korelirane, ter se v napovednem modelu želimo izogniti temu, da bi določene podatke "upoštevali večkrat". Omenjeno težavo sem poskušal rešiti s PCA, vendar se je izkazalo, da na znižanje RMSE predprocesiranje na ta način ni bistveno vplivalo. Zato sem se odločil za "ročno kalibracijo neodvisnosti". Da bomo to lahko storili, si oglejmo korelogram (graf korelacijske matrike). Opazimo, da na ceno diamanta najbolj vplivajo napovedne spremenljivke *carat*, *length*,

width in *depth*, vendar so omenjene napovedne spremenljivke med seboj tudi zelo korelirane. Po koreliranosti s ceno (kljub večjemu padcu korelacijskega koeficienta) sledita kategoriji *clarity* in *color*. Korelacijski koeficienti (s ceno) so dobri pokazatelj katere napovedne spremenljivke bomo bolj upoštevali pri napovedi cene, a si sedaj za vsako izmed napovednih spremenljivk oglejmo še napako, ki bi jo dobili, če bi za napoved uporabili le-to. Slednje nam omogoči funkcija *mean_error_mask*. Po izpisu povprečnih napak opazimo, da bi lahko napovedne spremenljivke po vrednostih povprečne RMSE na hitro razdelili v dve skupini. Tiste, katerih povprečna vrednost RMSE je manjša od 3000 (kot tudi z najvišjim korelacijskim koeficientom so to *carat*, *length*, *width* in *depth*), ter tiste, katerih povprečna vrednost RMSE je večja od 4000 (preostale napovedne spremenljivke). Opaženo nam še dodatno potrdi, katere napovedne spremenljivke bomo pri napovedu cene bolj/manj upoštevali.

Izračune RMSE, za vsako izmed napovednih spremenljivk, sedaj upoštevajmo za okvirno "mero pomembnosti" vsake izmed napovednih spremenljivk pri napovedi cene diamanta. V ta namen definirajmo *SCALE_0*, ki s skaliranjem pred iskanjem najbližjih sosedov zadošča za RMSE $751,934$.

Ideja nadaljevanja je sledeča - vzemimo *SCALE_0* za začetno vrednost funkcije *find_best_scale*. Ta za začetni vektor z naključnimi koraki (za izbran inkrement - pozicijo v vektorju, ter pozitivno ali negativno smer) išče vektor, ki (z algoritmom najbližjih sosedov pri nekem številu sosedov) vrne manjši RMSE. Če funkcija takšen vektor/inkrement najde, naredi "korak" v to smer oziroma obdrži nov vektor, drugače pa nadaljuje z naključnim iskanjem v istem stanju.

Začel sem z inkrementom $0,25$, ter ga nato pospotoma zmanjšal na $0,1$, $0,75$, $\ldots$, $0,005$ in $0,001$, ko algoritem v velikem številu korakov ni našel boljše izbire za vektor (takrat lahko trdimo, da za dani inkrement lokalno ni boljše izbire, ter se premaknemo na manjši inkrement). (Opomba: v resnici gre za nekakšno diskretno iskanje lokalnega minimuma, ter smiselno izbiro začetnega približka).

Na ta način sem našel *BEST_SCALE*, za katerega RMSE znaša le še $561,57$. Prileganje pri tako izbranem vektorju, je za prvih 200 diamantov vsakega izmed odsekov razvidno iz izrisanih grafov.

Omenil bi le še, da sem poskusil tudi z dodajanjem kvadratnih členov v model (saj bi z razširitvijo mojega prostora, vsaj v teoriji, lahko našel še manjšo napako). Slednje je preveč opočasnilo iskanje povprečne napake, ter iteracije funkcije *find_best_scale*, da bi lahko napako (v omejenem času) spravil pod zgoraj zapisano mejo.