Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Erik Štrumbelj

Učinkovita razlaga napovedi klasifikacijskih in regresijskih modelov

DOKTORSKA DISERTACIJA

Mentor: prof. dr. Igor Kononenko

Ljubljana, 2011

Povzetek

Razlaga modelov za napovedovanje je pomemben del procesa odkrivanja zakonitosti v podatkih, saj pripomore k hitrejšemu in boljšemu razumevanju modela ter poveča uporabnikovo zaupanje v napovedi modela. Osredotočili smo se na metode, ki vsakemu atributu pripišejo njegov prispevek k napovedi modela. Večina teh metod je prilagojenih za določen tip modela, nekatere izmed njih je moč uporabiti za razlago poljubnega modela, kar olajša uporabo in primerjavo različnih modelov.

Obstoječe splošne metode ne upoštevajo interakcij preko vseh podmnožic atributov, zato določenih primerov ne razložijo pravilno. V doktorskem delu predlagamo splošno metodo za računanje prispevkov posameznih atributov k napovedi modela, ki te interakcije upošteva in odpravi pomanjkljivosti obstoječih metod. Dokazali smo, da je metoda povezana s pojmom iz teorije iger – Shapleyevo vrednostjo. Eksponentno časovno zahtevnost, ki izhaja iz pregleda vseh elementov potenčne množice množice atributov, smo omilili z aproksimacijskim algoritmom. Učinkovitost algoritma dodatno izboljšamo s selektivnim in kvazi-naključnim vzorčenjem. Predlagali smo tudi mehnizem, ki uporabniku omogoča uravnavanje razmerja med časom računanja in pričakovano napako.

Praktično vrednost in učinkovitost predlagane metode smo pokazali na naboru umetnih in realnih množic podatkov in z uporabo večjega števila različnih klasifikacijskih in regresijskih modelov. Opisali smo aplikacijo metode pri napovedovanju ponovitve raka dojke, pri kateri so onkologi ocenili, da so razlage napovedi uporabne in v skladu z njihovim specialističnim znanjem. Navedemo tudi nekaj drugih uspešnih aplikacij predlagane metode. Izvedli smo tudi študijo, v kateri smo preverili, kako dobro se uporabniki učijo iz podanih primerov (brez ali z razlago) in napovedujejo vrednosti za nove, neznane primere. Rezultati pokažejo, da razlaga s prispevki atributov v povprečju pripomore k razumevanju problema oz. večji točnosti napovedi.

Ključne besede:

Odkrivanje zakonitosti v podatkih, interpretacija, vizualizacija, pomembnost atributov.

Abstract

Providing an explanation for prediction models is an important part of knowledge discovery. It helps with a quicker and better understanding of the model and increases the user's level of trust in the model's predictions. We focus on methods, which assign to each input feature its contribution to the model's prediction. Most such methods are model-specific. However, some can be used for any type of model, thus simplifying their use and enabling the comparison of different types of models.

Existing general methods do not take into account the interactions across all subsets of input features and in certain cases fail to provide a proper explanation. We propose a general method, which takes all interactions into account and deals with the shortcomings of existing methods. We prove that the proposed method is related to the Shapley value - a well-known concept from game theory. We deal with the resulting exponential time complexity by using an approximation. We use selective sampling and quasi-random sampling to further improve the efficiency of the approximation algorithm. We also propose a mechanism, which allows the user to select a tradeoff between the total running time and expected error.

Synthetic and real-world data sets are used and several different classification and regression models are applied to empirically show the practical utility of the proposed method. We describe how the method was applied to a real-world breast cancer recurrence problem and how oncologists confirmed the method's usefulness. We also list other successful application of the proposed method. We also conducted an experiment, during which we tested the users' ability to learn from examples (with or without an explanation) and make predictions for new and unknown instances. The results reveal that the explanation with contributions of input features help and increase the accuracy of the user's predictions.

Keywords:

Knowledge discovery in data, interpretation, visualization, feature importance.

IZJAVA O AVTORSTVU

doktorske disertacije

Spodaj podpisani Erik Štrumbelj,

z vpisno številko 63020161,

sem avtor doktorske disertacije z naslovom:

Učinkovita razlaga napovedi klasifikacijskih in regresijskih modelov.

S svojim podpisom zagotavljam, da:

- sem doktorsko disertacijo izdelal/-a samostojno pod mentorstvom
  prof. dr. Igorja Kononenka
- so elektronska oblika doktorske disertacije, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko doktorske disertacije
- in soglašam z javno objavo elektronske oblike doktorske disertacije v zbirki "Dela FRI".

V Ljubljani, dne 14.10.2011 Podpis avtorja:

Zahvala

Na prvem mestu bi se zahvalil svojemu mentorju prof. dr. Igorju Kononenku za vso potrpežljivost, nasvete in temelje, na katerih sem s svojim raziskovalnim delom lahko gradil naprej.

Posebno zahvalo si zaslužijo izr. prof. dr. Marko Robnik Šikonja, doc. dr. Matjaž Kukar, doc. dr. Zoran Bosnič, dr. Luka Šajn, mag. Petar Vračar, Darko Pevec, Ercan Canhasi in Domen Košir, sedanji in bivši člani Laboratorija za kognitivno modeliranje, za vso pomoč, nasvete in odlično delovno vzdušje.

Izvedba nekaterih raziskav ne bi bil mogoča brez izdatne pomoči onkologov prof. dr. Branka Zakotnika in dr. Cvetke Grašič Kuhar, za kar se jima iskreno zahvaljujem. Hvala dr. Marku Pregeljcu in prof. dr. Miranu Mihelčiču, ki sta mi omogočila uporabo razvitih metod na ekonomsko obarvanih problemih. Najlepše bi se zahvalil tudi Mirku Škofu, ki je vedno pripravljen razpravljati o matematiki in je bil v neprecenljivo pomoč pri marsikaterem matematičnem orehu.

Na koncu pa se globoko zahvaljujem vsem svojim bližnjim, še posebej staršem in Maji, ki me vseskozi spodbujajo in predvsem prenašajo.

Posvetilo

Kazalo

Poglavje 1

Uvod

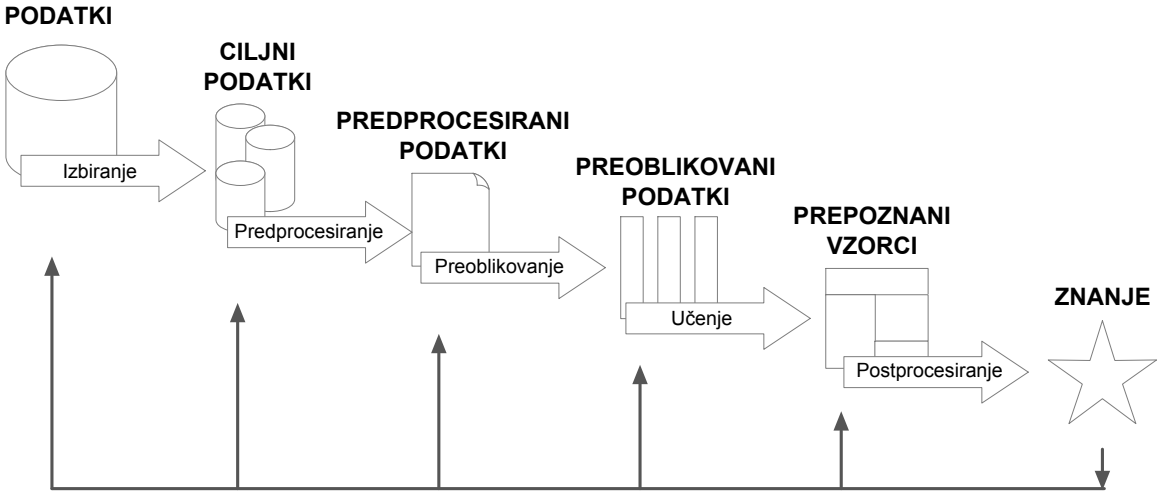
Problematika, s katero se bomo spopadli v tej doktorski disertaciji, spada na področje računalništva, natančneje, na področje odkrivanja zakonitosti v podatkih. To je interdisciplinarno področje, ki zajema korake od izbire podatkov, predprocesiranja in podatkovnega rudarjenja, vse do interpretacije podatkov (glej sliko 1.1). Osredotočili se bomo na proces interpretiranja vzorcev, ki jih dobimo iz podatkov z metodami podatkovnega rudarjenja (angl. data mining).

Glede na namen metode podatkovnega rudarjenja razdelimo na dve visokonivojski skupini: opisovanje in napovedovanje [29]. Pri opisovanju iščemo vzorce v podatkih, ki podatke opišejo na uporabniku razumljiv način. Pri napovedovanju iščemo vzorce, ki nam pomagajo opisati prihodnje izide nekih dogodkov. Najznačilnejša predstavnika prve skupine sta razvrščanje (angl. clustering) in povezovalna pravila. V drugo skupino, ki je za nas bolj zanimiva, pa spadata uvrščanje (od tu dalje klasifikacija) in regresija.

Pri klasifikaciji in regresiji vzorce opišemo z modelom, ki povezuje vhodne spremenljivke (od tu dalje atribute) z odvisno spremenljivko. Postopku, s katerimi zgradimo model, pravimo tudi učni algoritem, množici podatkov, ki jo pri tem uporabimo, pravimo učna množica. Procesu gradnje modela na učni množici z uporabo nekega učnega algoritma bomo krajše rekli kar učenje.

S pomočjo naučenega modela za nov primer atributov napovemo vrednost odvisne spremenljivke. Poleg napovedi nas pri napovedovanju zanima tudi narava povezav med vhodnimi in odvisno spremenljivko. Ni namreč nujno, da je model predstavljen na uporabniku razumljiv način. Razlaga teh povezav ima dvojen namen. Prvič, pomaga nam pri razumevanju delovanja modela. In drugič, razlaga modela, ki je uspešen pri napovedovanju, nam lahko ponudi dotlej neznano znanje.

Sedaj lahko natančno opredelimo področje te doktorske disertacije - razlaga napovedi klasifikacijskih in regresijskih modelov.



Slika 1.1: Stopnje procesa odkrivanja zakonitosti v podatkih [29].

1.1 Preprost ilustrativni primer

Preden formalno opredelimo najpomembnejše pojme, si poglejmo preprost primer. Na neki fakulteti je čas izpitov in ravno v tem trenutku tam potekajo ustni izpiti pri profesorju A. Prof. A je med študenti prvega letnika znan po zanimivi metodi ocenjevanja. Za vsakega študenta namreč pred ustnim izpitom vrže igralno kocko[1]. Če le na kocki ne pade šestica, je študent oproščen ustnega spraševanja in je izpit uspešno opravil. V nasprotnem primeru mu prof. A zastavi eno samo vprašanje, študentova usoda pa je odvisna od tega, kako dobro se je pripravil na izpit.

Študentje so hitro razpoznali ta koncept in se zelo dobro naučili napovedovati verjetnost, da bo nek študent opravil izpit. Tisti, ki se dobro naučijo celotno snov, bodo zagotovo uspešni. Tisti, ki se ne naučijo prav nič, imajo še vedno pet šestin možnosti, da izpit opravijo. Možnosti manj ekstremnih študentov pa se nahajajo nekje vmes, porazdeljene sorazmerno količini naučenega znanja[2].

Za študenta S_1, ki učbenika ni niti odprl, bil pa je vprašan, lahko hitro in pravilno napovemo, da so možnosti, da je izpit opravil enake 0. Ko bo tak študent prišel domov, bo njegove kolege zanimalo, *zakaj* je padel na izpitu. Študent bo najverjetneje razložil, da je imel izjemno smolo, da ga je profesor vprašal, malo pa si je kriv tudi sam, ker se ni vsaj malo učil. Po drugi strani pa bi študent S_2, ki zna vse in je bil prav tako vprašan, trdil, da je za visoko verjetnost uspeha zaslužen sam. To, da je bil vprašan, pa ni igralo pomembne vloge.

Kako se situacija spremeni, če profesorja A zamenjamo s profesorjem B, ki vedno sprašuje? Sam koncept ostane nespremenjen, saj študent še vedno pade le v primeru, ko

[1] Predpostavljamo, da sta poštena tako kocka kot profesor A.

[2] Lahko bi rekli, da so učno množico podatkov študentje sestavili z opazovanjem minulih izpitnih rokov pri profesorju A. Učni algoritmi, s katerimi so prišli do teh uporabnih vzorcev, pa so bili kar njihovi možgani.

je vprašan in ne zna odgovoriti na vprašanje. Napovedi za študenta S_1 in S_2, ki bosta oba vprašana, se ne spremenita. Prvi nima možnosti, drugi bo izpit zagotovo opravil. Spremeni pa se kontekst, saj imamo opravka s profesorjem, ki študente bolj pogosto sprašuje. Kaj se zgodi z razlago?

Za študenta S_2 bi še vedno rekli, da so njegove možnosti tako visoke, ker se je dobro pripravil. V primerjavi z izpitom pri profesorju A, je pri profesorju B pomembnost dobre pripravljenosti še večja, saj bo študent gotovo vprašan. Študent S_1 pa bi moral priznati, da si je za ničelne možnosti uspeha kriv povsem sam, ker se ni učil. Pomembna stvar, ki jo razberemo iz tega ilustrativnega primera, je, da razlaga napovedi ni odvisna samo od koncepta, ki ga razlagamo, temveč tudi od konteksta, v katerem razlagamo.

1.2 Formalna opredelitev ključnih pojmov

Opredelimo primer iz prejšnjega odstavka na formalen način. Osnovni element vsake napovedi je *primer*, za katerega napovedujemo vrednost odvisne spremenljivke. V našem primeru gre za študenta. Vsak primer ima eno ali več lastnosti, ki jih opišemo z atributi, skupaj z ostalimi primeri iste vrste pa tvori *prostor primerov*.

Definicija 1.1. *Naj bo $\mathcal{X} = X_1 \times X_2 \times ... \times X_n$ prostor primerov z n atributi, kjer je vsak atribut X_i neprazna množica s končno mnogo elementi.*

V našem primeru so to študentje. Posameznega študenta opišemo z dvema atributoma. Prvi opisuje količino znanja, drugi ali je bil vprašan. V skladu z definicijo 1.1 bi prostor študentov lahko opisali z

$$\mathcal{X}_{studenti} = \{0, 1, 2, 3, 4, 5\} \times \{\text{vprasan}, \text{ni vprasan}\},$$

kjer smo znanje opisali z lestvico od 0 do 5. Določen primer $\vec{x}_i \in \mathcal{X}_{studenti}$ predstavimo z vektorjem $\vec{x}_i = \langle x_{i,1}, x_{i,2} \rangle$. Kot je opisano kasneje, smo tu brez škode za splošnost privzeli, da so atributi končne množice. Notacijo $X_{i,j}$ pogosto uporabimo za označevanje j-te vrednosti i-tega atributa.

Glavni cilj pri napovedovanju je določanje vrednosti neke odvisne spremenljivke za nov (in morebiti neznan) primer iz množice primerov. V našem primeru je odvisna spremenljivka možnost študenta, da bo opravil izpit. Odvisna spremenljivka je z vrednostmi atributov povezana z nekimi koncepti. Te koncepte modeliramo s funkcijo $g : \mathcal{X} \to \Re$, ki ni nujno deterministična.

V našem primeru opis koncepta poznamo, v praksi pa je g običajno neznana funkcija. Na voljo imamo le množico r primerov, za katere smo izmerili vrednost odvisne spremenljivke:

$$\begin{aligned}
\vec{x}_1 &= \langle x_{1,1}, x_{1,2}, ..., x_{1,n} \rangle; g(\vec{x}_1) \\
\vec{x}_2 &= \langle x_{2,1}, x_{2,2}, ..., x_{2,n} \rangle; g(\vec{x}_2) \\
\vec{x}_3 &= \langle x_{3,1}, x_{3,2}, ..., x_{3,n} \rangle; g(\vec{x}_3) \\
&... \\
\vec{x}_r &= \langle x_{r,1}, x_{r,2}, ..., x_{r,n} \rangle; g(\vec{x}_r)
\end{aligned}$$

Z izbranim učnim algoritmom poskušamo v učni množici poiskati vzorce, ki nam bodo pomagali pri napovedovanju vrednosti g za prihodnje primere. Pri tem si pomagamo z morebitnim predznanjem o prostoru atributov in konceptih. Rezultat učenja je model f.

Definicija 1.2. *Model f neke neznane funkcije g nad prostorom atributov $\mathcal{X}$ preslika primer iz $\mathcal{X}$ na realno os (napoved)*

$$f_{\mathcal{X},g} : \mathcal{X} \to \Re.$$

V našem primeru so se študentje na podlagi preteklih izkušenj s svojim modelom približali skritemu opisu koncepta:

$$f_{\mathcal{X}_{studenti},g}(\vec{x}) \approx g(\vec{x}) = \begin{cases} 1 & x_2 = \text{ni vprašan} \\ \frac{x_1}{5} & x_2 = \text{vprašan} \end{cases},$$

kjer je $\vec{x} \in \mathcal{X}_{studenti}$ nek študent. Za model pogosto uporabimo poenostavljeno notacijo in namesto $f_{\mathcal{X},g}(\vec{x})$ pišemo samo $f(\vec{x})$.

Na tem mestu je potrebno pojasniti razliko med klasifikacijo in regresijo. S klasifikacijskimi metodami uvrščamo primere v eno izmed končno mnogo različnih kategorij (razredov). Z regresijskimi metodami pa modeliramo zvezno funkcijo. Napoved slednjih lahko opišemo z realnim številom. Klasifikacijsko napoved gledamo z vidika določene razredne vrednosti. Takšna napoved je bodisi binarna (primer pripada/ne pripada razredu) bodisi realna vrednost (verjetnost, da pripada razredu, ocena ali rang). V teoretičnem delu ne bomo razlikovali med klasifikacijskimi in regresijskimi modeli, temveč jih bomo obravnavali na enoten način.

Z vidika napovedovanja je model f vse, kar potrebujemo. Za razlago moramo, kot smo videli na primeru študenta S_1, opredeliti še pojem konteksta.

Definicija 1.3. *Kontekst $p_{\mathcal{X}}$ nad prostorom spremenljivk $\mathcal{X}$ je funkcija, za katero velja*

$$\sum_{x \in \mathcal{X}} p_{\mathcal{X}}(x) = 1 \text{ in } p_{\mathcal{X}}(x) \geq 0, \forall x \in \mathcal{X}.$$

Kontektst je porazdelitvena funkcija nad končno množico primerov.

Z našega profesorja A in njegove študente bi kontekst zapisali kot

$$p_A\left(\langle i, \text{ni vprašan} \rangle\right) = \frac{5}{6 \cdot 6} \text{ in } p_A(\langle i, \text{vprašan} \rangle) = \frac{1}{6 \cdot 6},\ i = 0..5,$$

Podobno bi kontekst izpita pri profesorju B določili z

$$p_B\left(\langle i, \text{ni vprašan}\rangle\right) = 0 \text{ in } p_B(\langle i, \text{vprašan}\rangle) = \frac{1}{6},\ i = 0..5.$$

Sedaj lahko definiramo tip razlage, ki je predmet našega raziskovanja.

Definicija 1.4. *Metoda za računanje prispevkov atributov k napovedi modela f za primer $\vec{x} \in \mathcal{X}$ v kontekstu p je preslikava*

$$\varphi : (f, \mathcal{X}, \vec{x}, p) \rightarrow \langle \varphi_1, \varphi_2, ..., \varphi_n \rangle$$

ki vsakemu izmed atributov X_i priredi prispevek $\varphi_i \in \Re$.

Z malo truda lahko naše intuitivne razlage napovedi za študenta z uporabo definicije 1.4 zapišemo tudi formalno. Brez škode se lahko dogovorimo, da bomo nepomembnim atributom dodelili prispevek 0. Pomembnim spremenljivkam dodelimo realno število, katerega velikost odraža pomembnost atributa. Da ne izgubimo informacije o tem, kateri atributi so za študenta ugodni, kateri pa neugodni, bomo za slednje uporabili negativen predznak, za ugodne pa pozitiven predznak.

Za napoved $f(\langle \text{vprašan}, 0\rangle) = 0$ za študenta S_1 pri profesorju B, bi prvemu atributu dodelili 0, saj vemo, da profesor vpraša vsakega študenta. Ta atribut ni pomemben. Drugemu atributu bi dodelili negativno vrednost, npr. -10, saj vemo, da je v tem primeru ravno študentova slaba pripravljenost kriva za slabe obete.

Pri profesorju A pa dejstvo, da je bil vprašan, igra pomembnejšo vlogo in ima negativen vpliv na študentove možnosti. Predpostavimo, da je enako pomembno kot dejstvo, da se ni učil, in obema atributoma pripišimo -5. Sedaj lahko razlagi primerjamo med seboj in ugotovimo, da je za izpit pri profesorju B učenje bolj pomembno kot za izpit pri profesorju A.

Seveda smo primera razložili bolj po občutku, a ideja razlage napovedi s prispevki atributov je jasna. Posameznemu atributu pripišemo vrednost, ki odraža njegovo pomembnost ter vpliv na napoved. Vrednosti naj bo moč primerjati z vrednostmi drugih atributov in preko različnih primerov.

1.3 Trije nivoji računanja prispevkov atributov

Preden lahko pojasnimo cilje tega doktorskega dela, moramo pojasniti nekatere razlike med metodami, ki atributom pripisujejo prispevke. Metode bomo razvrstili na 3 različne nivoje, glede na praktičen pomen prispevkov. Najvišji nivo predstavlja *splošni prispevek atributa* ali krajše, *pomembnost atributa*. Ta nivo označimo s simbolom Λ, s katerim bomo tudi kasneje označevali metode, ki jih uvrščamo v ta nivo. Pomembnost atributa Λ_i nam sporoča točno to, kar namiguje že ime - kako pomemben je atribut. Največje število metod spada v ta nivo. V našem ilustrativnem primeru s profesorjem B bi rekli, da je drugi atribut nepomemben ($\Lambda_2 = 0$), saj ima vedno enako vrednost. Prvi atribut pa je očitno pomemben, zato bi mu pripisali pozitivno pomembnost ($\Lambda_1 > 0$).

Definicija 1.5. *Metoda za računanje pomembnosti atributov za model f in prostor atributov $\mathcal{X}$ v kontekstu p je preslikava*

$$\Lambda : (f, \mathcal{X}, p) \rightarrow \langle \Lambda_1, \Lambda_2, ..., \Lambda_n \rangle$$

ki vsakemu izmed atributov X_i priredi pombembnost $\Lambda_i \in \Re$.

V nekaterih praktičnih scenarijih nam zadošča že pomembnost atributa. Takšen primer je izbira podmnožice atributov, kjer nas zanima le razvrstitev atributov glede na njihovo pomembnost. Pogosto pa nas zanima ne samo pomembnost atributa, temveč tudi kakšen vpliv na napovedi modela imajo njegove posamezne vrednosti. V našem primeru bi nas lahko kakšen študent vprašal: Se možnosti študenta, da opravi izpit, povečajo ali zmanjšajo, če se dobro nauči? Pomembnost atributa $\Lambda_1 > 0$ nam razkrije zgolj to, da je odgovor na to vprašanje pomemben pri napovedi, ne ponudi pa nam odgovora. Za odgovor na to vprašanje potrebujemo metodo, ki pripiše prispevek posameznim vrednostim atributa. Take metode, ki računajo *splošni prispevek vrednosti atributa*, označimo s ψ. Za razliko od pomembnosti atributa, pri splošnem prispevku vrednosti atributa igra pomembno vlogo tudi predznak. Na primer, v našem primeru bi vsaka dobra metoda morala ugotoviti, da več učenja vedno pozitivno vpliva na možnosti, da študent opravi izpit: $\psi_{1,5} > \psi_{1,4} > ... > \psi_{1,0}$

Definicija 1.6. *Metoda za računanje splošne pomembnosti vrednosti atributa za model f in prostor atributov $\mathcal{X}$ v kontekstu p je preslikava*

$$\psi_i : (f, \mathcal{X}, p) \rightarrow \langle \psi_{i,1}, \psi_{i,2}, ..., \psi_{i,|X_i|} \rangle$$

ki vsaki vrednosti j atributa X_i priredi prispevek $\psi_{i,j} \in \Re$.

Nivo ψ nam pove več kot Λ, a še vedno ne vsega. Pogosto namreč naletimo na situacijo, ko ima v določenem primeru vrednost atributa drugačen prispevek kot v povprečju[3].V splošnem se to zgodi takrat, ko dva ali več atributov sovpliva na napoved modela - pravimo tudi, da so v interakciji. Zato vpeljemo še tretji, najnižji nivo, *prispevek vrednosti atributa k napovedi za določen primer* (krajše *prispevek atributa k napovedi*). Slednji je za nas najbolj zanimiv in ga označimo s φ (glej definicijo 1.4). Prispevki, ki smo jih obravnavali v zadnjih dveh odstavkih poglavja 1.1, prav tako spadajo na ta nivo.

V prejšnjih treh odstavkih smo predstavili tri nivoje. Ti nivoji si sledijo po pomenu in nakazali smo, da iz prispevkov višjega nivoja ne izvemo vsega, kar nam ponudijo prispevki nižjega nivoja. Kaj pa obratno? Nam poznavanje najnižjega nivoja φ preko vseh primerov v nekem prostoru primerov omogoča, da rekonstruiramo tudi splošni prispevek vrednosti atributa. Kaj pa pomembnost atributa? A so v določenih primerih nivoji ekvivalentni?

[3]Predstavljajte si pacienta, ki boleha za boleznijo, ki jo običajno uspešno pozdravijo z določenim antibiotikom. Če napovedujemo bolnikovo zdravstveno stanje čez dva tedna, potem odločitev, da mu damo zdravilo, v povprečju pozitivno prispeva k njegovem zdravstvenem stanju. V konkretnem primeru, ko je pacient alergičen na zdravilo, pa je lahko prispevek močno negativen.

Na primer, če med atributi ni interakcij, bi pričakovali, da je prispevek atributa k napovedi kar enak splošnemu prispevku vrednosti, ki jo ima atribut. Odgovori na tu zastavljena vprašanja so pritrdilni, a od nas zahtevajo še nekaj dela, ki je opisano v 2 in 3. poglavju.

1.4 Izhodišče in cilj doktorskega dela

Do sedaj je bilo razvitih že mnogo metod, ki na takšen ali drugačen način računajo in pripisujejo prispevke posameznim atributom. Večina teh metod je prilagojenih določenim tipom modelov in jih ni moč uporabljati pri drugih tipih modelov. Nekatere metode pa so splošne in jih lahko uporabimo za poljuben model f. Pregled področja razlage napovedi modelov sledi v 2. poglavju, že na tem mestu pa lahko povemo, da imajo obstoječe splošne metode φ-nivoja določene pomanjkljivosti, ki zmanjšujejo njihovo uporabno vrednost. Med te metode spada tudi metoda za razlago klasifikatorjev, ki sta jo razvila Robnik Šikonja in mentor doktorskega dela Kononenko [82]. Ta raziskava je avtorju služila kot izhodišče za to doktorsko delo, saj ga je seznanila z glavnim problemom obstoječih splošnih metod (vključno z [82]), ki zaradi neupoštevanja interakcij med atributi ob prisotnosti leteh ne dajejo želenih rezultatov. V eksperimentalni del tega doktorskega dela smo vključili tudi 5 umetnih klasifikacijskih množic, ki se zgledujejo po množicah iz [82].

Glavni cilj doktorskega dela je razviti *splošno* metodo za računanje prispevka atributa k napovedi (glej definicijo 1.4) in jo uporabiti za razlago napovedi poljubnih regresijskih in klasifikacijskih modelov. Razvita metoda naj bo dobra, predvsem pa boljša od obstoječih metod. Ta ohlapna opredelitev kvalitete razlage od bralca zahteva dobro mero potrpežljivosti. V znanstveni literaturi namreč ne najdemo jasnih smernic za primerjavo takih metod na podlagi lastnosti, ki so skupne vsem metodam. V večini publikacij se uspešnost oz. kvaliteta metod meri posredno. Bodisi preko vizualnega preverjanja ter subjektivnih ocen uporabnikov bodisi preko merjenja uporabnosti prispevkov pri izbiri podmnožice najpomembnejših atributov. Skozi izpolnjevanje glavnega cilja si zastavimo tudi posplošitev in poenotenje obstoječih metod za računanje prispevka atributov.

Za ovrednotenje predlagane metode uporabimo dve skupini kriterijev. V prvo skupino uvrstimo objektivne kriterije, ki vključujejo dokazljive lastnosti in merljive lastnosti (npr. časi računanja). V drugo skupino uvrstimo vizualno preverjanje in subjektivne ocene uporabnikov. Dokažemo, da ima predlagana metoda želene lastnosti, ki jih obstoječe metode nimajo. S poskusi pokažemo, da je metoda primerna tudi za praktično rabo. Kot smo namignili v prejšnjem podpoglavju, se pri razvoju metode za nivo φ ne moremo izogniti vprašanju, kako združiti prispevke atributov k napovedi (φ) preko več primerov v splošne prispevke vrednosti teh atributov (ψ) in pomembnost atributov (Λ). Po utemeljitvi predlagane metode raziščemo tudi to vprašanje.

1.4.1 Prispevki k znanosti

- Razvoj splošne metode za razlago napovedi modelov s prispevki atributov v klasifikaciji in regresiji, ki upošteva tudi interakcije med atributi in s tem odpravi pomanjkljivost obstoječih metod. Metodo je v kombinaciji s poljubnim modelom moč uporabiti tudi kot filter-metodo za izbiro podmnožice atributov.
- Razvoj aproksimacijskega algoritma na osnovi vzorčenja, ki omili izhodiščno časovno zahtevnost predlagane metode in olajša praktično rabo. Algoritem je eksperimentalno ovrednoten na vrsti domen in z uporabo različnih tipov klasifikacijskih in regresijskih modelov.
- Razvoj ustrezne vizualizacije za razlago posamezne napovedi in modela kot celote.
- Uspešna aplikacija metode na problemu napovedovanja ponovitve raka dojke in študija praktične uporabnosti metode.

Del vsebine te doktorske disertacije je objavljen v zbornikih mednarodnih konferenc in znanstvenih revijah. Osnovna metoda je bila objavljena v [95] in [98]. Teoretična utemeljitev in aproksimacijski algoritem sta opisana v [96]. Aplikacija na problemu napovedovanja ponovitve raka dojke je opisana v [94]. Uporaba metode za razlago regresijskih modelov je opisana v [97], kjer najdemo tudi opis razlage celotnega modela.

Preostanek besedila je razdeljen na 5 poglavij. V naslednjem poglavju pregledamo širše področje razlage modelov v procesu odkrivanja zakonitosti v podatkih ter ožje področje računanja prispevkov atributov. V 3. poglavju najdemo opis predlagane metode in njeno teoretično utemeljitev, skupaj z izpeljavo aproksimacije, ki je potrebna za praktično rabo. Rezultati poskusov, ki smo jih izvedli na umetnih in realnih množicah podatkov, so opisani v 4. poglavju. Praktične aplikacije metode in študijo razumljivosti razlage najdemo v 5. poglavju. V 6. poglavju povzamemo ugotovitve in ponudimo nekaj idej za prihodnje delo.

Poglavje 2

Razlaga in razumljivost pri odkrivanju zakonitosti v podatkih

Obsežnost procesa odkrivanja zakonitosti v podatkih onemogoča jasno opredelitev vseh nalog, ki jih zajema. Vsekakor je ena izmed najpomembnejših nalog uporabniku ponuditi dobro razlago podatkov in znanje v razumljivi obliki. Ta naloga postaja vedno bolj pomembna, saj narašča število aplikacij odkrivanja zakonitosti oz. podatkovnega rudarjenja zunaj računalništva, na področjih kot so medicina, biologija, ekonomija, ipd. Cilj tega poglavja je bralcu ponuditi krajši pregled metod in tehnik, ki jih uporabljamo za pridobivanje bolj razumljivih rezultatov, ter poglobljeno analizo metod za računanje prispevkov atributov.

Poglavje je v grobem razdeljeno na dva dela. V prvem delu gremo bolj v širino in pogledamo metode in tehnike, ki se v procesu odkrivanja zakonitosti v podatkih uporabljajo za razlago in interpretacijo modelov. Pokažemo, da obstaja vrsta različnih načinov za povečanje razumljivosti rezultatov, ki niso omejeni samo na stopnjo postprocesiranja. V drugem delu poglavja se osredotočimo na metode za računanje prispevkov atributov. Te metode bolj podrobno analiziramo in izpostavimo njihove prednosti in slabosti. Ugotovitve iz tega poglavja nam v naslednjem poglavju pomagajo pri razvoju nove metode.

Proces odkrivanja zakonitosti v podatkih se običajno začne z neprocesiranimi podatki (glej sliko 1.1). V naslednjem koraku izločimo podatke, ki niso relevantni za izpolnjevanje zadane naloge. Ker so podatki pogosto nekonsistentni, vsebujejo šum ali celo manjkajo, se poslužujemo različnih metod predprocesiranja. Šele nato podatke transformiramo v obliko, ki je primerna za naslednji korak - korak učenja. Rezultati koraka učenja oz. podatkovnega rudarjenja pogosto niso primerni za končnega uporabnika, zato sledi še korak postprocesiranja rezultatov. Slednji je skoraj v celoti namenjen razlagi oz. povečevanju razumljivosti rezultatov, zato večina metod, ki jih omenimo v tem poglavju, spada v ta korak. Vendar metode razlage niso omejene le na postprocesiranje. Boljši razumljivosti se lahko približamo že zgodaj v procesu, če temu ustrezno predprocesiramo in transformiramo podatke, kasneje pa še z izbiro bolj transparentnih modelov.

2.1 Predprocesiranje

Največ metod, ki jih uporabljamo v koraku predprocesiranja, pripada skupini metod za izbiro podmnožice atributov. Običajno je namen uporabe takih metod izločiti nepomebne atribute in izbrati podmnožico atributov, ki, skupaj z uporabljenim modelom, dajo najboljše rezultate. Pogosto želimo zmanjšati razsežnost prostora atributov tudi z namenom zmanjšanja časov računanja ali porabe pomnilniškega prostora. Skupaj z izbiro podmnožice atributov obravnavamo tudi transformacijo prostora atributov. Prostor atributov lahko preslikamo v nov prostor atributov, ki je manj razsežen, bolj primeren za učenje in/ali ima druge želene lastnosti.

Čeprav razlaga ni osnovni namen omenjenih metod, jih lahko uporabimo za povečanje razumljivosti končnih rezultatov. Z velikostjo izbrane množice atributov lahko iščemo ravnotežje med kvaliteto rezultatov in končno velikostjo (zapletenostjo) modela. Posredno vplivamo tudi na razumljivost rezultatov, saj so krajše hipoteze oz. manj zapleteni modeli za uporabnika bolj razumljivi. Podobno lahko prostor atributov transformiramo v takega, v katerem lahko rezultate predstavimo na uporabniku bolj razumljiv način.

Metode izbire podmnožice atributov v grobem delimo na filtre, metode z zunanjo optimizacijo in metode z notranjo optimizacijo [33]. Filtri so metode, ki podmnožico atributov izberejo neodvisno od kasneje uporabljenega učnega algoritma. Metode z zunanjo optimizacijo obravnavajo učni algoritem oz. model kot črno škatlo, ter izbiro podmnožice optimizirajo glede na uspešnost napovedi [48]. Pri metodah z notranjo optimizacijo pa atribute izbiramo med samim procesom učenja. Za globlje razumevanje so na voljo številne knjige [34, 59, 60] in pregledni članki [21, 27, 33], ki so v celoti posvečene izbiri in transformaciji atributov.

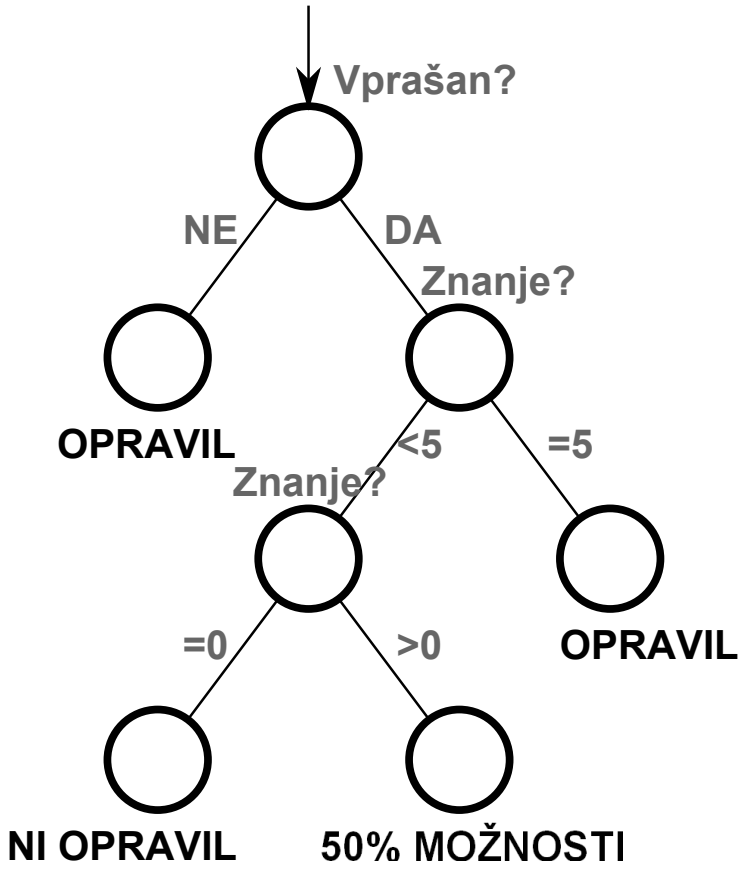
Podobno kot izbiramo najprimernejšo podmnožico atributov, lahko izberemo tudi najprimernejšo množico primerov, čeprav se slednje uporablja redkeje. Razlogi, zakaj bi želeli uporabiti le podmnožico primerov in ne vseh, so podobni kot pri izbiri podmnožice atributov. Z manjšo množico primerov zmanjšamo čas in prostor, ki je potreben za izvajanje učnih algoritmov. To je še posebej koristno pri algoritmih, ki temeljijo na učnih primerih, kot naprimer metode najbližjih sosedov. Prav tako z načrtnim odstranjevanjem določenih primerov zmanjšamo šum v podatkih. Posredno z zmanjševanjem števila primerov povečujemo razumljivost modelov. Za globlje razumevanje izbire podmnožice primerov glejte pregledne članke in knjigo [31, 43, 61, 62, 77].

2.2 Izbira transparentnih modelov

Klasifikacijski (regresijski) modeli se med seboj razlikujejo glede na način predstavitve naučenega znanja. Nekateri modeli znanje predstavijo na abstrakten in uporabniku nerazumljiv način. Za takšne modele pravimo, da so *netransparentni*. Kot pokažemo v sledečem podpoglavju, se za dosego razumljivosti znanja pri takih modelih poslužujemo metod postprocesiranja. Nekateri modeli pa že zaradi svoje zasnove znanje predstavijo

na uprabniku razumljiv, *transparenten* način, zato postprocesiranje ni nujno potrebno. Nekaj takih modelov predstavimo v tem podpoglavju.

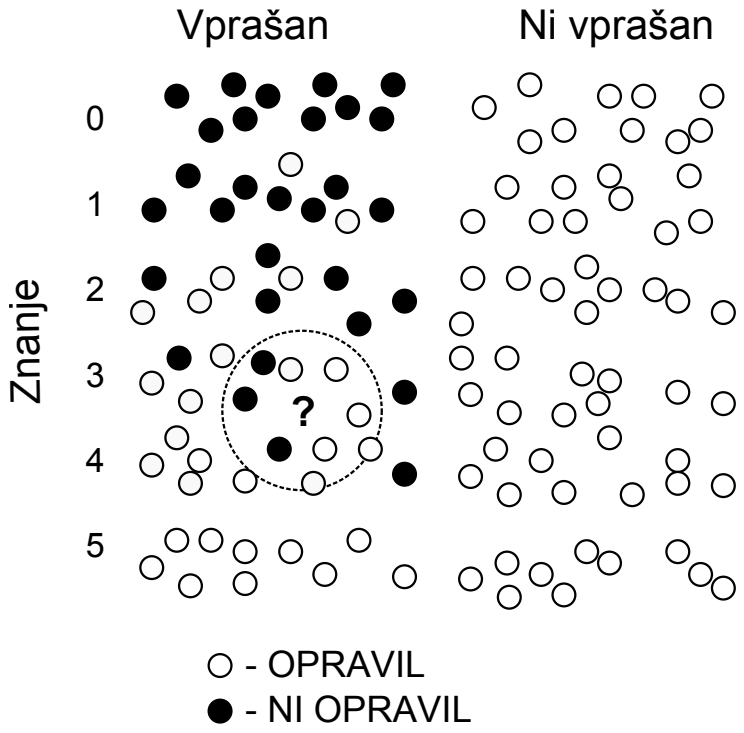
Eden najbolj znanih in razširjenih tipov transparentnih modelov, tudi na drugih področjih znanosti, so **odločitvena drevesa**. Na sliki 2.1 je primer odločitvenega drevesa, ki opisuje naš uvodni primer s profesorjem in študenti. Iz modela je razvidno, da vsak študent, ki se nauči prav vse (znanje = 5), izpit zagotovo opravi. Prav tako izpit opravijo študentje, ki niso vprašani. Tisti, ki se ne učijo, pa so vprašani, zagotovo padejo na izpitu. Preostali vprašani študenti pa imajo v povprečju 50% možnosti, da izpit opravijo. Poleg splošne razlage celotnega modela, ki nam jo ponuja slika 2.1, lahko razložimo tudi posamezen primer. To storimo tako, da se sprehodimo od korena do lista, v katerem leži naš primer, ter si zabeležimo pravilo, ki smo ga pri tem uporabili. Velja omeniti, da se razumljivost odločitvenih dreves manjša s številom vozlišč, število vozlišč pa potencialno narašča hitreje kot število atributov.



Slika 2.1: Odločitveno drevo za uvodni primer s študentom in profesorjem.

Tudi napovedi **metod najbližjih sosedov** so do neke mere transparentne. Vizualizacijo napovedi v eno-, dvo- ali trirazsežnem prostoru lahko uporabimo kot razlago. Na sliki 2.2 najdemo tako vizualizacijo za primer s profesorjem in študentom. S povečevanjem števila atributov ali primerov postane tak način razlage hitro nerazumljiv. Težave

odpravimo ali vsaj omilimo z izbiro ustrezne podmnožice najpomembnejših atributov in primerov. Nekatere metode za izbiro podmnožice primerov so bile razvite posebej za metode najbližjih sosedov [14, 76].



Slika 2.2: Primer za metodo najbližjih sosedov.

Iz statistike poznamo **linearni regresijski model**

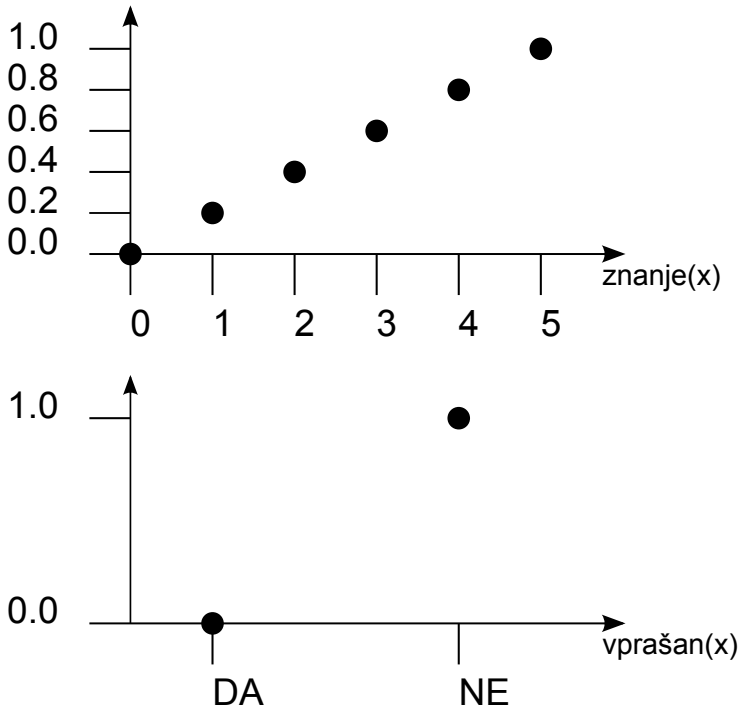
$$f(\vec{x}) = \beta_1 x_1 + \beta_2 x_2 + ... + \beta_2 x_n + \beta_0.$$

Ta tip modela ima zelo enostaven opis, saj je enolično določen s koeficienti $\beta_0, ...\beta_n$. Obenem preprosto izračunamo vpliv atributov v določenem primeru, saj vrednost atributa samo pomnožimo s pripadajočim koeficientom. Celotna napoved pa je vsota posameznih prispevkov in konstantnega člena β_0.

Podobne lastnosti ima **aditivni model**, ki je posplošitev linearnega modela

$$f(\vec{x}) = \sum_{i=1}^{n} f_i(x_i) + \beta_0,$$

kjer je f_i funkcija i-tega atributa. Tudi pri aditivnem modelu lahko, če poznamo funkcije f_i, prispevek i-tega atributa k napovedi zapišemo kot vrednost f_i v točki atributove vrednosti. Prav tako z vizualizacijo vseh f_i v celoti opišemo delovanje modela f (glej sliko 2.3). Četudi za nek model f ne poznamo funkcij f_i, vemo pa, da je aditiven, lahko za razlago modela uporabimo eno izmed metod postprocesiranja, ki so bile razvite posebej za aditivne modele.



Slika 2.3: Primer za aditivni model.

Včasih odvisne spremenljivke ne moremo modelirati z aditivnim modelom, lahko pa jo modeliramo, če uporabimo neko vezno funkcijo h:

$$f(\vec{x}) = E[g(\vec{x})] = h^{-1}(\sum_{i=1}^{n} f_i(x_i) + \beta_0),$$

pri čemer od h zahtevamo monotonost in odvedljivost. Takim modelom pravimo **posplošeni aditivni modeli** [38]. Z vidika prispevkov atributov za posplošene aditivne modele velja enako kot za aditivne modele.

Med bolj znanimi klasifikacijskimi modeli najdemo tudi **naivni Bayesovski klasifikator** (od tu dalje naivni Bayes). Pridevnik izvira iz predpostavke o pogojni neodvisnosti atributov pri dani razredni vrednosti, zaradi katere lahko napoved za razredno vrednost C zapišemo kot:

$$P(C|X_1...X_n) = P(C)\prod_{i=1}^{n}\frac{P(C|X_i)}{P(C)}. \tag{2.1}$$

Napovedani razred je tisti razred C, pri katerem je napovedana verjetnost največja. Kljub predpostavki o pogojni neodvisnosti je ta model zelo uporaben, še posebej na področju medicinske diagnostike [51].

Dodatno prednost modela opazimo, če logaritmiramo desno stran enačbe (2.1), saj tako namesto produkta dobimo vsoto členov, ki predstavljajo informacijske prispevke posameznih atributov [7, 49]. V takšni obliki je naivni Bayes (posplošeni) aditivni model in ga lahko na tak način tudi razlagamo.

Linearni regresijski model, aditivni model in naivni Bayes so modeli, pri katerih napoved razlagamo s prispevki atributov k napovedi, celoten model pa s splošnimi prispevki vrednosti atributov. Spadajo v skupino metod in tehnik, ki je za nas še posebej zanimiva, zato jih ponovno in bolj podrobno obravnavamo v podpoglavju 2.4.

2.3 Postprocesiranje

V prejšnjem podpoglavju smo si ogledali nekaj transparentnih modelov, pri katerih je znanje že samo po sebi podano na uporabniku razumljiv način. Transparentnost je obenem tudi posledica slabosti, saj so v primerjavi z netransparentnimi modeli ti zaradi večje preprostosti v povprečju manj uspešni pri napovedovanju. V poskusu, da bi združili uspešnost zapletenejših modelov in razumljivost transparentnih modelov, so razvili več metod postprocesiranja. S temi metodami iz netransparentnih modelov izluščimo uporabniku razumljivo znanje, a ohranimo uspešnost pri napovedovanju.

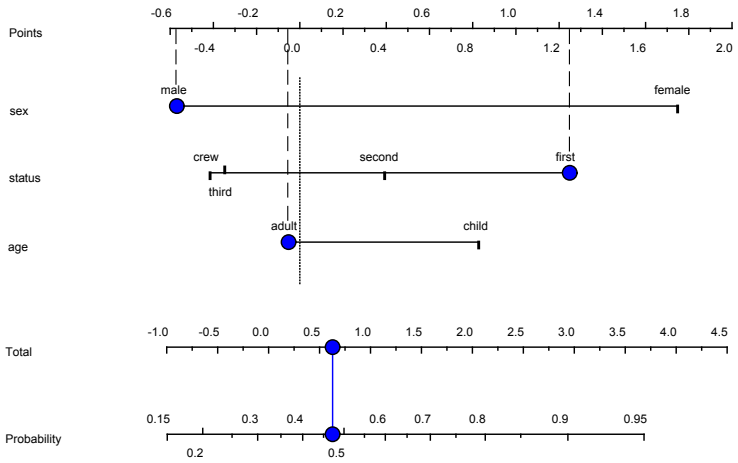
Metode v grobem razdelimo v dve skupini. V prvi so metode, ki so prilagojene določenemu tipu modela in izkoriščajo njegove lastnosti. V drugi skupini pa splošne metode, ki jih uporabimo za poljuben model oz. večjo družino modelov (regresijski modeli, klasifikacijski modeli).

2.3.1 Postprocesiranje za določene tipe modelov

V prejšnjem podpoglavju smo omenili, da je bilo razvitih več metod za razlago napovedi **aditivnih modelov**, ki izkoriščajo lastnost, da napoved aditivnega modela napišemo kot vsoto prispevkov posameznih atributov. Poulin in sod. [88] predlagajo postopek ExplainD za razlago naivnega Bayesa, SVM z linearnim jedrom in logistične regresije. Predlagan postopek izkorišča prej omenjene lastnosti aditivnih modelov in ga lahko razširimo tudi na druge aditivne modele. Podoben pristop uporabijo Možina in sod. za naivnega Bayesa [68] ter Jakulin in sod. za SVM z linearnim jedrom [42], pri čemer za vizualizacijo prispevkov uporabijo nomograme (glej sliko 2.4). Nomograme so uporabili tudi v medicinskih aplikacijah za vizualizacijo modela logistične regresije [63] in Coxovega modela [37, 44].

Nomogram je dvorazsežni diagram, s pomočjo katerega lahko uporabnik grafično določi vrednost funkcije v določeni točki (glej sliko 2.5). Začetki nomografije segajo v 19. stoletje, njihov glavni namen pa je bil inženirjem in drugim uporabnikom omogočiti hitro in natančno računanje vrednosti zapletenih funkcij. Izraz nomogram ni povsem korekten opis za vizualizacije, ki jih uporabljajo prej omenjene metode, saj mora uporabnik odčitane prispevke posameznih atributov še vedno sešteti. Pravi nomogram za primer iz uvoda najdemo na sliki 2.6. Da jih razlikujemo od nomogramov, bomo takim vizualizacijam pravili kvazi-nomogrami[1]. Bralci, ki jih nomografija še posebej zanima, naj si preberejo Doerflerjev pregled snovanja nomogramov [28].

[1]Uporaba kvazi-nomogramov namesto pravih je razumljiva, saj zapletenost in površina nomograma,



Slika 2.4: Kvazi-nomogram izdelan s pomočjo orodja Orange [26].

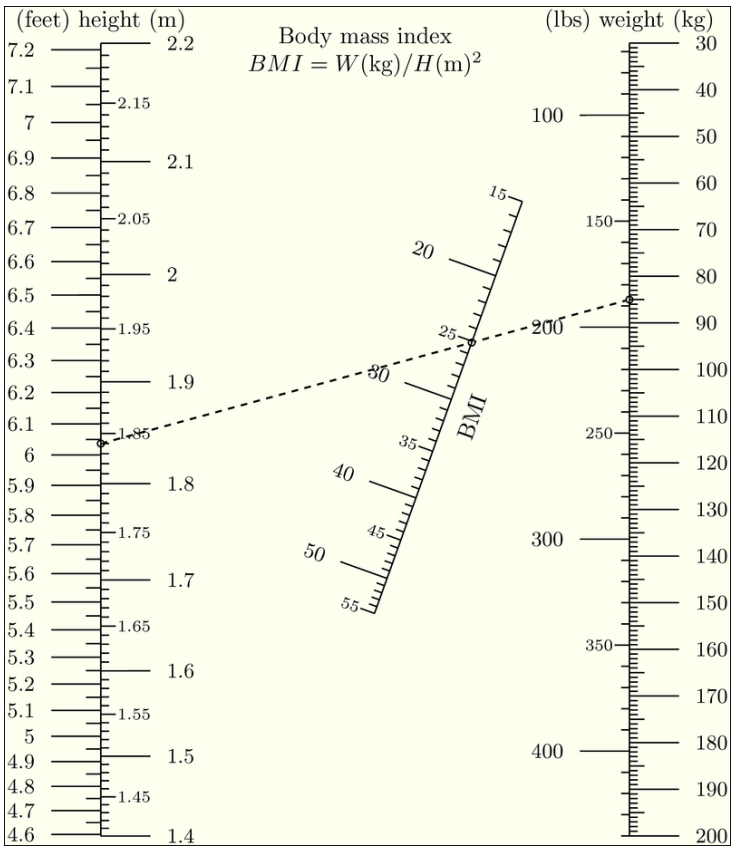
Eden najuspešnejših tipov modelov, pa tudi eden izmed najmanj transparentnih, so **umetne nevronske mreže**. Posledično so raziskovalci na več različnih načinov poskušali izluščiti bolj razumljivo znanje iz umetnih nevronskih mrež [4, 20, 39, 52, 69, 90, 101]. Podobno velja za metodo podpornih vektorjev ali **SVM** (Support Vector Machine) [6, 36, 42, 65, 72, 91, 93, 102].

Kljub temu, da so učni algoritmi, ki znanje opišejo s pravili (**odločitvena pravila**, **drevesa**), v osnovi bolj transparentni, razumljivost njihovih hipotez pada z naraščanjem števila pravil (ali velikostjo drevesa). V ta namen je bilo razvitih več metod, ki implementirajo mehanizem za iskanje ravnotežja med velikostjo množice pravil in kvaliteto in/ali postprocesirajo množico generiranih pravil [3, 9, 8, 10, 11, 12, 15, 22, 24, 25, 75, 84, 100].

Metode, ki izboljšajo razumljivost naučenega znanja so bile razvite tudi za nekatere druge transparentne modele, naivnega Bayesa z boostingom [78] in Bayesovske mreže [64, 23]. Breimanovi učni algoritem naključni gozdovi (Random forest) [13] prav tako omogoča računanje pomembnosti atributov. Pomembnost atributa pri določenem drevesu in za določen primer je definirana kot razlika med točnostjo drevesa in povprečno točnostjo drevesa, če permutiramo vrednosti atributa in s tem izgubimo informacijo, ki jo nosi ta atribut. Če slednje povprečimo preko vseh primerov in vseh dreves, dobimo splošno pomembnost atributa. Strobl in sod. [85, 86] ponudijo dodatno izboljšavo s pogojenimi permutacijami, ki izboljša osnoven izračun pomembnosti atributa. Potrebno je poudariti, da pri naključnih gozdovih govorimo o pomembnosti atributa za točnost modela in ne o pomembnosti atributa za model[2].

ki omogoča povsem grafično računanje, naraščata s številom atributov.

[2]V enem primeru opazujemo vpliv atributov na točnost, v drugem pa vpliv atributov na napoved modela.



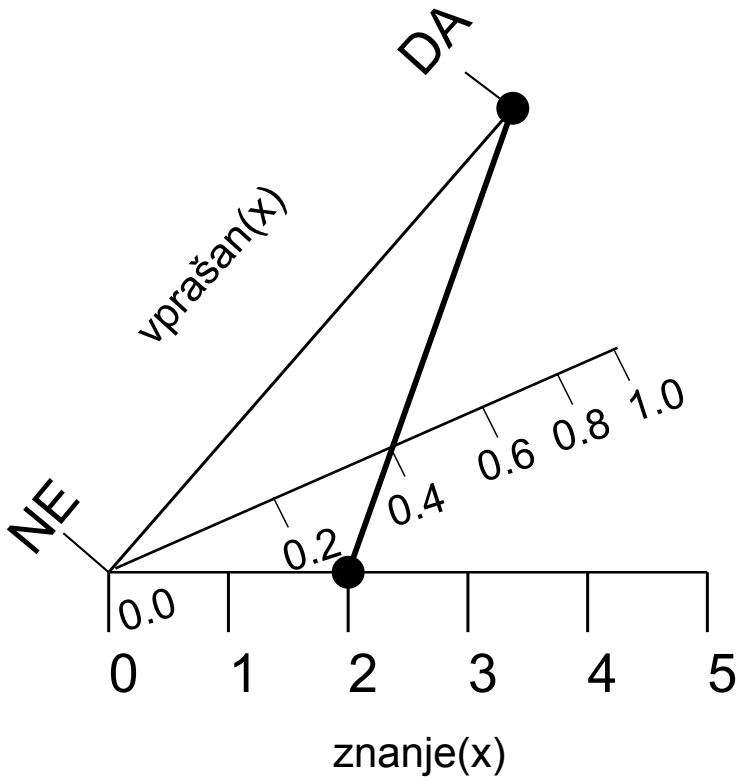
Slika 2.5: Ilustrativni primer nomograma za izračun indeksa telesne mase (www.pynomo.org).

2.3.2 Univerzalne metode

Uporaba metod, ki smo jih omenili do tega trenutka, je odvisna od izbire modela, ki ga uporabljamo. Obstajajo tudi univerzalne metode, ki jih lahko uporabimo za razlago oz. izboljšanje razumljivosti poljubnega klasifikacijskega in/ali regresijskega modela. Univerzalne metode imajo v primerjavi s specifičnimi določene prednosti. Omogočajo enostavnejšo primerjavo različnih modelov strojnega učenja. Prav tako lahko na enoličen način odkrivamo napake v modelih.

Z vidika strokovnjaka s področja odkrivanja zakonitosti v podatkih so univerzalne metode zaželene, saj ni potrebno implementirati različne metode za vsak tip modela posebej. Z vidika končnega uporabnika rezultatov in napovedi metod podatkovnega rudarjenja univerzalne metode razlage nudijo to prednost, da jih ni potrebno zamenjati, če zamenjamo tip modela. Uporabnikova interakcija z napovedmi in razlago ostane nespremenjena, uporabnik pa prihrani na času in trudu, ki bi ga namenil prilagajanju novemu tipu razlage.

Osnovni pogoj, ki ga mora izpolnjevati vsaka univerzalna metoda, je, da jo je moč na enoličen način uporabiti za poljuben model f. Ne smemo uporabiti specifičnih lastnosti



Slika 2.6: Pravi nomogram.

modelov, temveč jih moramo obravnavati kot črne škatle (t.j. neznane funkcije f). Vse kar nam preostane, je spreminjanje primera na vhodu in opazovanje sprememb v napovedi modela. Najpreprostejši način bi bil napovedati vse primere in s tem dobiti popolno sliko funkcije f. Ker je vseh primerov v prostoru atributov ponavadi zelo veliko (pogosto neskončno mnogo), smo prisiljeni sklepati kompromise med številom primerov, ki jih bomo uporabili za razlago modela, in časovno zahtevnostjo postopka razlage. Kot je opisano kasneje v tem poglavju, ti kompromisi privedejo do situacij, ko določena metoda razlage odpove (t.j., napačno razloži prispevek k napovedi ali pomembnost atributa).

Najbolj razširjen način ugotavljanja pomembnosti nekega atributa za model f je t. i. **analiza občutljivosti** modela na spremembe vrednosti tega atributa. Prispevek i-tega atributa k napovedi modela f za primer $\vec{x}$ je običajno definiran kot razlika med napovedjo modela za primer in pričakovano napovedjo, če vrednosti i-tega atributa ne poznamo:

$$\text{marginal}\varphi_i = f(\vec{x}) - E[f(\vec{x} \setminus x_i)]. \tag{2.2}$$

Enačbi (2.2) pravimo tudi **marginalni (robni) učinek** i-tega atributa. Ta pristop k univerzalni razlagi sta uporabila Robnik Šikonja in Kononenko [82] pa tudi Lemaire in sod. [56].

Idejo lahko posplošimo tudi na računanje pomembnosti atributa v celoti. Poglejmo si na primeru pred kratkim objavljene metode **FIRM** (Feature Importance Ranking

Measure) [103]. Najprej definirajmo splošni prispevek j-te vrednosti i-tega atributa kot pričakovano napoved, če ima i-ti atribut ravno j-to vrednost:

$$\text{FIRM}\psi_{i,j} = E[f(\vec{x}|x_i = \mathcal{X}_{i,j}]. \tag{2.3}$$

Z uporabo enačbe (2.3) definiramo pomembnost i-tega atributa. Slednja je definirana kot standardna deviacija splošnih prispevkov vrednosti i-tega atributa:

$$\text{FIRM}\Lambda_i = \sqrt{\text{Var}_j[\text{FIRM}\psi_{i,j}]}. \tag{2.4}$$

Enačba (2.4) že govori o pomembnosti atributa v celoti in ne več o njegovem prispevku v določenem primeru. Spomnimo se, da nas v tem doktorskem delu zanima predvsem prispevek atributov k napovedi in delovanje modela, ne pa pomembnost atributov.

Naj poudarimo, da vse zgoraj naštete univerzalne metode razlagajo model oz. njegove napovedi s prispevki posameznih atributov. Omeniti velja tudi postopek, ko najprej z manj transparentnim modelom napovemo izide za množico primerov, nato pa s transparentnim modelom modeliramo zvezo med primeri in napovedmi prvotnega modela. Pri tem lahko uporabimo poljuben transparenten model. S tem dobimo posredno transparentno razlago napovedi manj transparentnega modela. Ta postopek se sicer redko uporablja, saj je transparentni model običajno manj kompleksen in ne more modelirati kompleksnejšega znanja manj transparentnega modela. Ravno to pa nas najbolj zanima.

2.4 Razlaga s prispevki atributov

Do sedaj smo v tem poglavju napravili pregled metod in tehnik, ki služijo povečevanju razumljivosti skozi različne korake procesa odkrivanja zakonitosti v podatkih. Sedaj si ponovno in bolj temeljito poglejmo metode, ki nas najbolj zanimajo. To so metode, ki računajo prispevke atributov, s katerimi lahko tudi razlagamo delovanje modela. Pri razlagi so nam v pomoč metode vseh treh nivojev Λ, ψ in φ.

Pregled širšega področja je pokazal, da so prispevki atributov pogost pristop k razlagi. Skupaj z ekstrakcijo logičnih pravil gre pravzaprav za dva najpogostejša pristopa k razlagi. Če vzamemo samo univerzalne metode, pa večina metod temelji ravno na računanju prispevkov atributov.

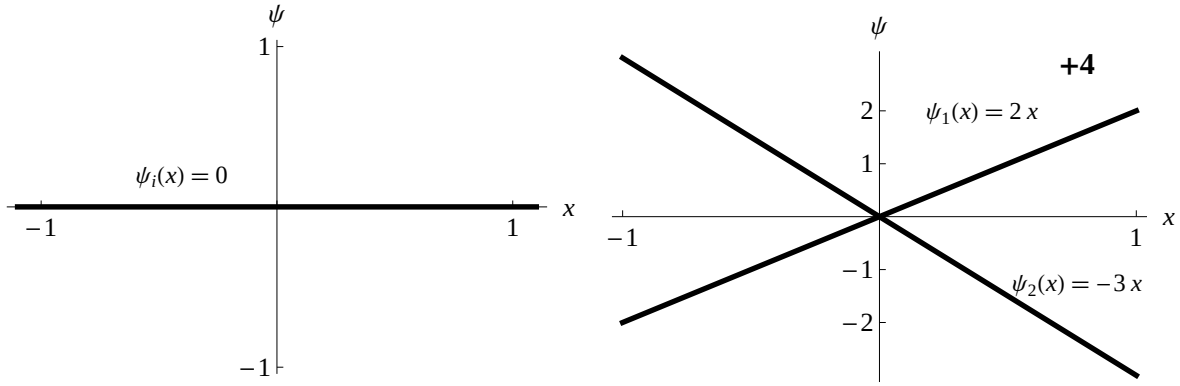
2.4.1 Konstantni model

Začnimo z izjemno preprostim namišljenim učnim algoritmom, ki naučeno znanje opisuje le z modeli oblike

$$f(\vec{x}) = \beta_0,$$

kjer je $\beta_0 \in \Re$ poljubna konstanta. S takšnim učnim algoritmom se ne moremo veliko naučiti in v praksi bi bil neuporaben. Po drugi strani pa na enostaven način razložimo

vse modele, ki jih generira, saj nobeden izmed atributov ne vpliva na model. Prispevek i-tega atributa je $\varphi_i = 0$, neodvisno od modela, atributa, primera in izbire konteksta, v katerem razlagamo. Podobno je prispevek poljubne vrednosti i-tega atributa $\psi_i(x) = 0$, in pomembnost atributov $\Lambda_i = 0$. Prispevek atributa k neki napovedi in njegovo pomembnost opišemo z eno samo realno vrednostjo. Prispevek vrednosti i-tega atributa v odvisnosti od njegove vrednosti je funkcija $\psi_i : X_i \leftarrow \Re$, ki jo prikažemo z grafom (glej sliko 2.7(a)).



(a) $f(\vec{x}) = 0$ (b) $f(x_1, x_2) = 2x_1 - 3x_2 + 4$

Slika 2.7: Vizualizaciji splošnih prispevkov vrednosti atributov, iz katerih lahko rekonstruiramo napoved modela za poljuben primer.

2.4.2 Linearni model

Obstaja skupina učnih algoritmov (npr. algoritmi, ki generirajo samo zgoraj opisane konstantne modele), katerih modele lahko v celoti razložimo s splošnimi prispevki vrednosti. Računanje prispevkov postane težja naloga, ko preidemo na bolj zapletene modele. V naslednjem koraku si poglejmo lineare modele oblike

$$f(\vec{x}) = \beta_n x_n + \beta_{n-1} x_{n-1} + ... + \beta_1 x_1 + \beta_0,$$

kjer $\beta_i \in \Re$.

Za razliko od konstantnega modela, kjer potrebujemo samo eno vrednost, linearni model opišemo z n vrednostmi. V primerjavi z večino modelov, ki jih generirajo algoritmi podatkovnega rudarjenja, je ta zapis preprost. Posledično bi pričakovali, da je enako preprosto tudi določiti pomembnost posameznih atributov, a temu ni tako.

Vprašajmo se, kako pomemben je atribut X_1 za zgornji model. Achen [1] opiše več različnih načinov, na katere raziskovalci definirajo pomembnost atributa pri linearnih modelih. Prvi, **teoretična pomembnost** (angl. theoretical importance) predstavlja neposredno velikost spremembe napovedi, če spremenimo vrednost atributa. Potencial

spremembe, ki jo lahko povzroči i-ti atribut, je zajeta v koeficientu β_i. Teoretična pomembnost je neodvisna od konteksta in za nas neuporabna, saj z njo posledično ne moremo dobro razložiti model v različnih kontekstih[3].

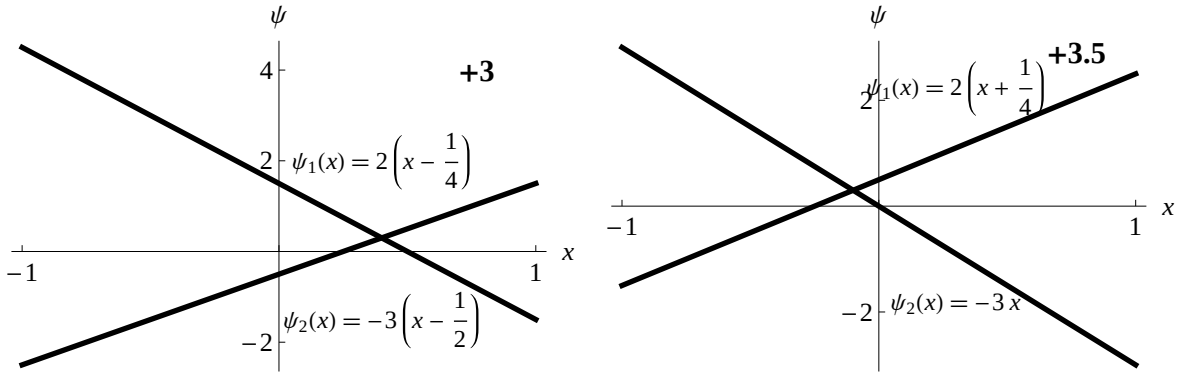
Drugi nivo pomembnosti, **situacijska pomembnost** (angl. level importance), predstavlja pomembnost atributa v konkretni situaciji (kontekstu), in je opredeljen kot $\beta_i E[X_i]$ (pričakovana vrednost člena v danem kontekstu). Vidimo, da je vsota situacijskih pomembnosti atributov in β_0 enaka pričakovani napovedi modela.

Achen predlaga, da prispevek vrednosti v določenem primeru $\vec{x}$ zapišemo kot

$$\text{linear}\psi_i(x_i) = \beta_i x_i - \beta_i E[X_i] = \beta(x_i - E[X_i]).$$

Gre za razliko od situacijske pomembnosti, ki jo povzroči vrednost atributa v tem določenem primeru. Podobno velja, da je vsota prispevkov vrednosti v določenem primeru in β_0 enaka napovedi modela za ta primer. Zapomnimo si, da je prispevek atributa v določenem primeru, ko ima ta vrednost x_i, enak povprečnemu prispevku vrednosti linear$\psi_i(x_i)$. To je posledica tega, da linearni model ne vsebuje interakcij med atributi, kar v tem delu srečamo še večkrat.

Za primer vzemimo prostor atributov $[-1,1] \times [-1,1]$ in model $f(\vec{x}) = 2x_1 - 3x_2 + 4$. Slika 2.7(b) je vizualizacija prispevka členov prvega in drugega atributa. Pozor, zgolj z uporabo te vizualizacije in β_0 lahko rekonstruiramo napoved modela za poljuben primer. Le odčitamo in seštejemo prispevka vrednosti obeh atributov ter prištejemo β_0. Vizualizacija v celoti razloži delovanje linearnega modela.



(a) $f(x_1, x_2) = 2x_1 - 3x_2 + 4$ (b) $f(x_1, x_2) = 2x_1 - 3x_2 + 4$

Slika 2.8: Vizualizaciji splošnih prispevkov vrednosti atributov, iz katerih lahko rekonstruiramo napoved modela za poljuben primer. Vizualizaciji sta za dva različna konteksta $E[X_1] = \frac{1}{4}, E[X_2] = \frac{1}{2}$ (a) in $E[X_1] = -\frac{1}{4}, E[X_2] = 0$ (b).

Kako se na vizualizaciji odraža sprememba konteksta? Najprej se spomnimo, da lahko pri linearnem modelu brez škode za splošnost privzamemo, da so vsi atributi centrirani

[3]Porazdelitev vrednosti atributov oziroma kontekst lahko močno vpliva na vrednost člena $\beta_i X_i$. Pomembnost upoštevanja konteksta pri razlagi smo pokazali že v uvodnem poglavju.

(t.j. $E[X_i] = 0, \forall i$), potrebna razlika pa je zajeta v β_0. Izkoristimo to dejstvo in pri danem kontekstu p preoblikujmo model f v

$$f'(\vec{x}) = \beta_n(x_n - E_p[X_n]) + \beta_{n-1}(x_{n-1} - E_p[X_{n-1}]) + ... + \beta_1(x_1 - E_p[X_1]) + \beta_0 + \sum_i \beta_i E_p[X_i] =$$

$$= \beta_n(x_n - E_p[X_n]) + \beta_{n-1}(x_{n-1} - E_p[X_{n-1}]) + ... + \beta_1(x_1 - E_p[X_1]) + E_p[f(\vec{x})],$$

pri čemer smo uporabili zgoraj ugotovljeno, da je pričakovana napoved vsota situacijskih prispevkov in β_0. Vrnimo se na primer $[-1, 1] \times [-1, 1]$, $f(\vec{x}) = 2x_1 - 3x_2 + 4$ in kontekst p_0, da je $E_{p_0}[X_1] = \frac{1}{4}, E_{p_0}[X_2] = \frac{1}{2}$. Pričakovana napoved je $E_{p_0}[f] = \frac{1}{2} - \frac{3}{2} + 4 = 3$, na sliki 2.8(a) pa najdemo vizualizacijo, ki je z vidika rekonstrukcije napovedi ekvivalentna tisti na sliki 2.7(b).

Sedaj poglejmo isti primer, a spremenimo kontekst v tak kontekst p_1, da je $E_{p_1}[X_1] = -\frac{1}{4}, E_{p_1}[X_2] = 0$. Dobimo $E_{p_1}[f] = -\frac{1}{2} - 0 + 4 = 3\frac{1}{2}$. Vizualizacija na sliki 2.8(b) je razlaga prispevkov posameznih vrednosti v kontekstu p_1. Kot pokaže primerjava s kontekstom p_0 na sliki 2.8(a), je pri linearnih modelih razlika med konteksti le v razliki med pričakovano napovedjo v enem in drugem kontekstu, funkcije splošnih prispevkov vrednosti posameznih atributov pa se ne spremenijo. Če se za konec vrnemo na prvi primer, vidimo, da je slika 2.7(b) vizualizacija splošnih prispevkov vrednosti modela v kontekstu $E[X_1] = E[X_2] = 0$.

Opisana metoda je v praksi učinkovit način za računanje prispevkov atributov in razlago linearnih modelov.

2.4.3 Aditivni model

Aditivni model je podoben linearnemu, pri čemer je napoved vsota funkcij posameznih atributov

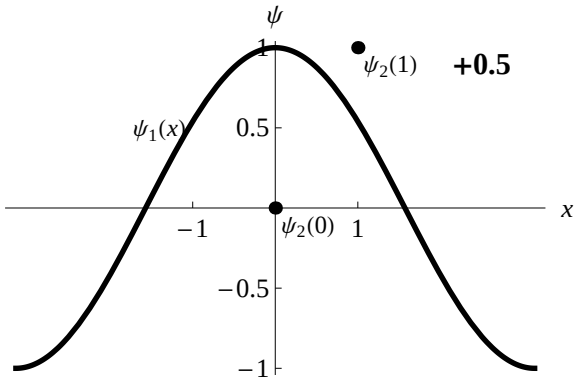
$$f(\vec{x}) = \sum_{i=1}^{n} f_i(x_i) + \beta_0.$$

Podobno kot pri linearnemu modelu, tudi pri aditivnemu modelu centriramo posamezne funkcije atributov:

$$f'(\vec{x}) = \sum_{i=1}^{n} (f_i(x_i) - E[f_i]) + \sum_{i=1}^{n} (E[f_i]) + \beta_0.$$

Lahko uporabimo enako vizualizacijo kot pri linearnih modelih, pri čemer splošni prispevek vrednosti atributa v odvisnosti od njegove vrednosti ni več nujno linearna funkcija.

Vzemimo primer prostora atributov $[-\pi, \pi] \times 0, 1$, model $f(\vec{x}) = cos(x_1) + x_2$ in tak kontekst p_0, da je X_1 enakomerno porazdeljena na $[-\pi, \pi]$ in $E_{p_0}[X_2] = \frac{1}{2}$. Sledi $E_{p_0}[f] = \frac{1}{2}$, slika 2.9 pa je ustrezna vizualizacija razlage tega modela v kontekstu p_0.



Slika 2.9: Vizualizacija splošnih prispevkov vrednosti atributov, iz katerih lahko rekonstruiramo napoved modela $f(x_1, x_2) = cos(x_1) + x_2$.

2.4.4 Naivni Bayes

Kot smo že omenili, lahko model naivnega Bayesa

$$P(C|X_1...X_n) = P(C)\prod_{i=1}^{n}\frac{P(C|X_i)}{P(C)} \tag{2.5}$$

zapišemo v aditivni obliki, z logaritmiranjem obeh strani enačbe:

$$-log_2P(C|X_1...X_n) = -log_2P(C) - \sum_{i=1}^{n}(log_2P(C|X_i) - log_2P(C)). \tag{2.6}$$

Vsota na desni strani enačbe 2.6 je vsota informacijskih prispevkov atributov [49]. Tudi informacijski prispevek je funkcija vrednosti atributa, zato lahko enačbo 2.6 preoblikujemo v ustrezno obliko aditivnega modela ter model vizualiziramo na enak način, kot prej aditivne modele. Na podlagi istega načela, z nekoliko drugačnimi vizualizacijami, so naivnega Bayesa razlagali v različnih študijah [7, 49, 68]. Na medicinskih domenah se je razlaga s prispevki izkazala kot uspešna in razumljiva za medicinske strokovnjake [51].

2.4.5 Univerzalni pristopi - marginalni učinek atributa

Za razliko od prej opisanih metod, pri univerzalnih pristopih ne poznamo lastnosti modela f. Najbolj pogost pristop določanja prispevka atributov je uporaba marginalnega prispevka atributa. Ponovimo definicijo:

$$\text{marginal}\varphi_i = f(\vec{x}) - E[f(\vec{x} \setminus x_i)]. \tag{2.7}$$

Ta pristop je znan tudi kot t.i. one-factor-at-a-time, saj pomembnost vsakega atributa obravnavamo posebej, brez spreminjanja vrednosti preostalih atributov.

Intuitivno pristop izgleda dobro - pomembnost atributa je razlika, ki jo dobimo, če ta atribut skrijemo. Če je pozitivna, je atribut očitno prispeval k povečanju f, če je

negativna, pa k zmanjšanju napovedi. Večja ko je sprememba, bolj pomembna je vrednost atributa v tem primeru. Ničelni prispevek pa je ekvivalenten nepomembnemu atributu. Slednja trditev drži samo v eno smer. Če je atribut nepomeben (t.j. ne nastopa v neznani funkciji f oz. je njegova vloga v praksi zanemarljiva), potem bo očitno tudi prispevek enak nič, ne glede na primer $\vec{x}$. Obratno pa ne velja.

Protiprimere, ko pristop odpove, najdemo v sitacijah, ko atributi sovplivajo (so v interakciji). V osnovi marginalni prispevek upošteva tudi interakcije, saj s permutacijo razbijemo tudi morebitne sovplive z drugimi atributi, kar pa ne deluje v vseh primerih. Poglejmo si primer $\mathcal{X} = \{0,1\} \times \{0,1\}, f(\vec{x}) = x_1 \vee x_2$ in razložimo napoved $f(1,1) = 1$ v kontekstu, kjer so vse kombinacije atributov enako verjetne. Ker sprememba ene same vrednosti ne spremeni napovedi, saj je ena enica dovolj, velja $\varphi_1 = \varphi_2 = 0$. Oba atributa obveljata za nepomembna, čeprav je jasno, da oba igrata vlogo pri napovedi.

Omenjeni primer izpostavi največjo pomanjkljivost uporabe marginalnih prispevkov za razlago. Pomanjkljivost ni omejena samo na disjunkcijo, temveč povzroča težave kadar koli nastopi redundanca med atributi. Da bi v takih primerih pravilno ocenili pomembnost atributov, bi morali hkrati "skriti" vse take atribute, da bi se pokazala razlika v napovedi. V naslabšem primeru je potrebno preiskati vse podmnožice atributov!

Pristop ima tudi nekaj želenih lastnosti. Poleg univerzalnosti, je učinkovit, saj obravnavanje vsakega atributa posebej nima visoke časovne zahtevnosti. Poleg tega lahko pokažemo, da je v primeru aditivnega modela f pristop ekvivalenten prej omenjeni razlagi za aditivne modele. Vsoto vseh prispevkov za primer lahko zapišemo kot

$$\sum_{i=1}^{n} \text{marginal}\varphi_i = \sum_{i=1}^{n} (f(\vec{x}) - E[f(\vec{x} \setminus i)]) =$$

$$= n(f(\vec{x}) - (n-1)\sum_{i=1}^{n}(f_i(x_i) + E[f_i]) = f(\vec{x}) - E[f],$$

kar je enako razliki med napovedjo in pričakovano napovedjo. Tudi marginalni učinki marginalφ predstavljajo dekompozicijo napovedi modela. To je pomembna prednost, saj v primeru aditivnega modela ni potrebno zamenjati metode razlage, ampak je rezultat ekvivalenten uporabi prej omenjene razlage za aditivne modele.

2.4.6 Univerzalni pristopi - FIRM

Metodo FIRM [103] smo že omenili in opisali v enačbah (2.3) in (2.4). Osnovno idejo lahko opišemo na preprost način. Najprej za vsako vrednost i-tega atributa zabeležimo pričakovano napoved ob pogoju, da ima i-ti atribut to vrednost. Nato izračunamo varianco teh vrednosti. Če je atribut pomemben, potem bodo njegove vrednosti povzročile spremembe na izhodu (in obratno).

Čeprav je metoda namenjena računanju pomembnosti atributa za model (Λ-nivo), lahko pogojno napoved FIRM$\psi_i(x)$ uporabimo za razlago prispevka te vrednosti atributa

(glej enačbo (2.3)). Če pogledamo primer, ki smo ga uporabili pri pristopu z marginalnimi učinki, vidimo, da je FIRM$\psi_1(1) = 1$, FIRM$\psi_1(0)$ pa je manjša od 1. Vrednosti atributa povzročijo spremembo in končni prispevek (glej enacbo (2.4)) bo večji od 0.

FIRM nima pomanjkljivosti, ki je značilna za pristope z marginalnimi učinki, vendar odpove na drugem tipu situacij. Poglejmo si primer $\mathcal{X} = \{0,1\} \times \{0,1\}, f(\vec{x}) = x_1 \text{xor} x_2$ v kontekstu, kjer so vse kombinacije atributov enako verjetne. Sledi FIRM$\psi_1(1) = \frac{1}{2}$ in FIRM$\psi_1(0) = \frac{1}{2}$, saj vrednost prvega atributa sama po sebi ne pove nič o napovedi - potrebni sta obe vrednosti. Posledično je v tem primeru FIRM$\Lambda_1 =$ FIRM$\Lambda_2 = 0$, kar je v nasprotju z dejstvom, da sta oba atributa pomembna.

Ekskluzivni-ali je učinkovit protiprimer, a obenem umeten in redko prisoten v realnih podatkih. Pomanjkljivost ni omejena samo na ekskluzivni ali, temveč na vse situacije, kjer en atribut igra vlogo "stikala" in obrne vlogo nekega drugega atributa. Praktične primere najdemo v medicini. V začetni fazi nekaterih bolezni je za pacientovo pričakovano življenjsko dobo bolje, če je mlad in vitalen, saj njegovo telo bolje prenaša bolezen in posledice zdravljenja. Ko bolezen napreduje preko določene meje, pa je za pričakovano življenjsko dobo bolje, če je človek starejši, saj bolezen v mlajšem telesu hitreje napreduje. Podoben primer najdemo tudi v športu, na primer nogometu ali košarki. Ko je ekipa v prednosti, bo zavlačevanje igre povečalo možnost, da ekipa zmaga. Ko pa je ekipa v zaostanku, bo zavlačevanje igre zmanjšalo možnost zmage.

2.5 Pomembna spoznanja

Ponovimo nekaj najpomembnejših ugotovitev, do katerih smo prišli v tem poglavju:

- Uporaba prispevkov atributov je pogost pristop k razlagi, pri univerzalnih metodah je najpogostejši.
- Največ metod se posveča Λ-nivoju (pomembnost atributov), kamor spadajo tudi metoda za izbiro podmnožic atributov, najmanj metod pa φ-nivoju, ki je z vidika razlage najbolj uporaben.
- Pri modelih, ki ne vsebujejo interakcij med atributi (aditivni modeli), so prispevki nivojev ψ in φ ekvivalentni. Z drugimi besedami, če ni interakcij med atributi, potem je prispevek neke vrednosti atributa k poljubni napovedi enak splošnemu prispevku te vrednosti.
- Univerzalne metode za razlago so pomanjkljive. Za odpravo pomanjkljivosti je potrebno upoštevati vse podmnožice atributov.
- Računanje prispevkov atributov pri (posplošenih) aditivnih modelih je dobro raziskano, a metode niso poenotene, temveč so prilagojene posameznim tipom aditivnih modelov.

Poglavje 3

Posplošen prispevek atributa k napovedi

V tem poglavju opišemo predlagano metodo za razlago s prispevki posameznih atributov. Za osnovo nam služi spoznanje iz prejšnjega poglavja, da je potrebno upoštevati potencialne interakcije med vsemi možnimi podmnožicami atributov. Pokažemo, da je metoda dobro utemeljena in ima želene lastnosti, vendar tudi eksponentno časovno zahtevnost, ki omejuje praktično rabo. V ta namen predlagamo aproksimacijski algoritem, ki omili časovno zahtevnost.

Predlagano razlago s prispevki atributov prilagodimo za računanje splošnega prispevka vrednosti nekega atributa in splošnega prispevka spremenljivke, ki lahko služi tudi kot ocena pri izbiri podmnožice spremenljivk. Na koncu poglavja dokažemo, da predlagana metoda posploši razlago s prispevki za aditivne modele. Najprej pa sledi definicija pojmov iz teorije iger, ki so nam v pomoč pri utemeljevanju metode.

3.1 Pojmi iz teorije iger

Spoznajmo neantagonistično koalicijsko igro za n igralcev in koncept rešitve − Shapleyevo vrednost. Bralci, ki so s temi pojmi že seznanjeni, lahko to podpoglavje mirno preskočijo.

Definicija 3.1. *Koalicijska igra za n igralcev je dvojec $\langle N, v\rangle$, kjer je $N = \{1, 2, ..., n\}$ končna množica n igralcev in $v : 2^N \to \Re$ karakteristična funkcija, za katero velja $v(\varnothing) = 0$.*

Podmnožicam množice N pravimo tudi koalicije, celotni množici N pa velika koalicija vseh igralcev. Karakteristična funkcija opisuje vrednost posamezne koalicije. Pri koalicijski igri običajno privzamemo, da se je velika koalicija uspešno formirala. Vprašanje pa je, kako na pošten način razdeliti njeno vrednost med vseh n udeležencev, saj so nekateri izmed njih lahko h koaliciji prispevali več kot drugi. Rešitev koalicijske igre je operator ϕ, ki igri $\langle N, v\rangle$ pripiše vektor plačil $\phi(v) = (\phi_1, ..., \phi_n) \in \Re^n$.

Praktičen primer koalicijske igre nastane, ko skupina n posameznikov oropa banko. Po uspešnem ropu si poskušajo na pošten način razdeliti naropano vsoto $v(N)$. Pri tem morajo upoštevati, koliko bi uspele naropati posamezne podmnožice celotne skupine $(v(S), S \subseteq N)$. Z drugimi besedami, pri delitvi morajo upoštevati doprinos posameznikov.

Za vsako igro z vsaj enim igralcem obstaja neskončno mnogo rešitev[1]. Med temi rešitvami pa so nekatere bolj poštene kot druge. In čeprav imamo vsi neko intuitivno predstavo o tem, kakšna je poštena delitev, jo težje formalno definiramo. Elegantno rešitev je ponudil Shapley [83], ki je poštenost aksiomatiziral s sledečimi štirimi aksiomi:

Aksiom 1. $\sum_{i \in N} \phi_i(v) = v(N)$ *(učinkovitost).*

Aksiom 2. *Če za igralca i in j za vsak S velja $v(S \cup \{i\}) = v(S \cup \{j\})$, kjer je $S \subset N$ in $i, j \notin S$, potem $\phi_i(v) = \phi_j(v))$ (simetričnost).*

Aksiom 3. *Če za igralca i za vsak $S \subset N \setminus \{i\}$ velja $v(S \cup \{i\}) = v(S)$, potem $\phi_i(v) = 0$ (slamnati igralec).*

Aksiom 4. *Za poljuben par iger v, w velja $\phi(v + w) = \phi(v) + \phi(w)$, kjer je $(v + w)(S) = v(S) + w(S)$ za vsak S (aditivnost).*

Prvi izmed štirih aksiomov pravi, da mora biti vsota izplačil enaka skupni vrednosti velike koalicije. V kontekstu bančnega ropa bi to pomenilo, da si morajo roparji razdeliti nič več in nič manj kot vsoto plena. Aksiom simetričnosti primerja vpliv para igralcev na podmnožico preostalih igralcev. Če imata igralca vedno enak vpliv na podmnožico, potem naj dobita enako izplačilo. Z drugimi besedami, če sta dva roparja prispevala enako, je pošteno, da oba dobita enak delež. Tretji aksiom pravi, da igralec, ki ne prispeva nič, ne dobi plačila. Četrti aksiom pa govori o aditivnosti rešitve preko različnih iger. Predstavljamo si, da skupina roparjev v letu izvede več ropov. Če je postopek delitve aditiven, potem je vseeno, če si plen razdelijo po vsakem ropu posebej ali celoten plen na koncu leta. V obeh primerih bi moral vsak ropar dobiti enak delež.

Izrek 3.2. *Za koalicijsko igro $\langle N, v \rangle$ obstaja enolična rešitev ϕ, ki zadosti aksiomom 1 do 4.*

$$Sh_i(v) = \sum_{S \subseteq N \setminus \{i\}, s=|S|} \frac{(n-s-1)!s!}{n!}(v(S \cup \{i\}) - v(S)), \quad i = 1, ..., n.$$

Dokaz. Dokaz izreka najdemo v [83]. □

Shapley je poiskal rešitev koalicijske igre, ki zadosti štirim aksiomom (glej Izrek 3.2). Ta rešitev se, njemu v čast, imenuje Shapleyeva vrednost. Najbolj presenetljiv rezultat Shapleyevega dela pa je ugotovitev, da štirje aksiomi omejijo prostor možnih rešitev na

[1] Niso pa vse rešitve tudi smiselne. Rešitev, da si ropar, ki ukrade le 10 €, dodeli milijon €, je v teoriji sprejemljiva, a v praksi neuporabna.

eno samo. Z drugimi besedami, Shapleyeva vrednost je edini način delitve, ki zadosti tem štirim aksiomom.

Shapleyeva vrednost se uporablja na širokem spektru različnih področij, kar podpira tudi pregledni članek Morettija in Patroneja [67], ki je v celoti posvečen Shapleyevi vrednosti in aplikacijam. Aplikacijo Shapleyeve vrednosti najdemo tudi na področju odkrivanja zakonitosti v podatkih. Kienan in sod. [45] so jo uporabili pri lokalizaciji funkcije bioloških in umetnih omrežij. Njihovo delo so kasneje prilagodili za izbiro podmnožice atributov [18], pri čemer so bile podmnožice atributov koalicije, njihova vrednost pa točnost pripadajočega modela.

Sorodne metode, ki temeljijo na uporabi Shapleyeve vrednosti, najdemo tudi na podpodročju statistike, ki se ukvarja s pomembnostjo spremenljivk (angl. relative variable importance) [92]. Metode za računanje pomembnosti vhodnih spremenljivk pri linearnih regresijskih modelih se v osnovi ukvarjajo s problemom dekompozicije variance modela med posamezne vhodne spremenljivke [32]. Dva najmodernejša pristopa sta PMVD [30] in LMG [17]. Slednji je ekvivalenten Shapleyevi vrednosti igre, kjer koalicije predstavljajo podmnožice atributov, vrednost koalicije pa je kvaliteta modela (R^2). Pristop, ki je bil znan že nekoliko prej, je leta 1987 obudil Kruskal [53, 54]. Stufken je leta 1992 pripomnil, da je LMG [17] ekvivalent Shapleyevi vrednosti [87]. Kasneje pristop odkrijejo še enkrat, izhajajoč iz Shapleyeve vrednosti [58].

3.2 Napoved kot vsota vseh interakcij med atributi

Naj bo $\mathcal{X}$ prostor n atributov, p kontekst, $\vec{a} \in \mathcal{X}$ primer, ki ga razlagamo, in f model. Množica $N = \{1, ..., n\}$ naj bo množica indeksov n atributov. Ker želimo splošno metodo, ne predpostavimo drugih lastnosti funkcije f.

V prejšnjem poglavju smo videli, da vse splošne metode opazujejo razliko v napovedi, če vrednost nekega atributa "skrijemo". V takem primeru lahko vrednost atributa opišemo s slučajno spremenljivko, napoved pa zapišemo z matematičnim upanjem. Z drugimi besedami, na posamezen primer bomo gledali kot na dogodek. Za verjetnostno funkcijo bomo uporabili porazdelitveno funkcijo konteksta p.

Naš glavni cilj je izračunati prispevek, ki ga ima atribut k napovedi za nek primer $\vec{a}$. V praksi se prispevki atributov k napovedi manifestirajo kot razlika v napovedi med pričakovano napovedjo in napovedjo, ko so znane vrednosti atributov za primer $\vec{a}$:

$$E[f|N] - E[f|\varnothing],$$

kjer $E[f|Q]$ uporabimo za zapis pogojnega matematičnega upanja

$$E[f|Q] = \sum_{\vec{x} \in \mathcal{X}; \forall i: x_i = a_i \vee i \notin Q} p'(\vec{x}) f(\vec{x}).$$

Razliko v napovedi, ki jo povzroči poznavanje vrednosti atributov, lahko posplošimo na poljubno podmnožico atributov.

$$\Delta(S) = E[f|S] - E[f|\varnothing], \forall S \subseteq N. \tag{3.1}$$

Spomnimo se pomembne ugotovitve iz drugega poglavja. Če želimo odpraviti pomanjkljivosti obstoječih metod, moramo upoštevati interakcije med vsemi možnimi podmnožicami atributov. Naj bo razlika, ki jo atributi povzročijo pri napovedi za primer $\vec{a}$, vsota interakcij vseh podmnožic:

$$\Delta(N) = \sum_{Q \subseteq N} \mathcal{I}(Q),$$

kar lahko posplošimo na poljubno podmnožico atributov

$$\Delta(S) = \sum_{Q \subseteq S} \mathcal{I}(Q). \tag{3.2}$$

Pri tem funkcijo interakcije $\mathcal{I}$ nismo definirali neposredno, temveč posredno. Lahko jo namreč izrazimo z razlikami Δ s preoblikovanjem enačbe (3.2):

$$\mathcal{I}(S) = \Delta(S) - \sum_{Q \subset S} \mathcal{I}(S), \forall S \subseteq N. \tag{3.3}$$

Iz enačbe (3.1) je razvidno, da je $\Delta(\varnothing) = 0$. Sledi $\mathcal{I}(\varnothing) = 0$. Skupaj s tema začetnima vrednostima enačbo (3.3) uporabimo kot rekurzivno definicijo interakcij. Enačba (3.2) za določeno $\Delta : \mathcal{P}(N) \to \Re$ enolično določi tudi interakcije $\mathcal{I} : \mathcal{P}(N) \to \Re$.

Sedaj razdelimo vrednost vsake interakcije med atribute, ki v njej sodelujejo, da dobimo prispevke posameznih atributov

$$\varphi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{\mathcal{I}(S \cup \{i\})}{|S \cup \{i\}|}, \quad i = 1, 2, ..., n. \tag{3.4}$$

S tem smo definirali našo metodo računanja prispevkov.

3.2.1 Povezava s Shapleyevo vrednostjo

Iz enačbe (3.4) je razvidno, da ima neposreden izračun prispevka eksponentno časovno zahtevnost. Drugo pomembno praktično vprašanje je izračun členov Δ. Preden se lotimo praktičnih vidikov, bomo dokazali izrek, ki povezuje predlagano metodo s Shapleyevo vrednostjo.

Izrek 3.3. *$\langle N = \{1, 2, ..., n\}, \Delta\rangle$ je koalicijska igra in $\varphi(\Delta) = (\varphi_1, \varphi_2, ..., \varphi_n)$ je enaka Shapleyevi vrednosti $Sh(\Delta)$.*

Za izrek 3.3 ponujamo dva dokaza. Prvi dokaz je bolj neposreden, saj preverimo, da predlagana funkcija izpolnjuje vse štiri Shapleyeve aksiome. Drugi dokaz je posreden, saj rekurzivno definicijo prispevka preobrazimo v eksplicitno, ki se izkaže za identično Shapleyevi vrednosti.

Dokaz 1. Aksiom 1: Pokazati moramo, da je $\sum_{i\in N}\varphi_i = \Delta(N)$. Naša osnovna ideja je bila, da je razlika, ki jo povzroči neka množica atributov, enaka vsoti interakcij vseh podmnožic te množice atributov (glej enačbo (3.2)). Interakcije smo nato v celoti razdelili med prispevke. V nasprotno smer: $\sum_{i=1}^{n}\varphi_i = \sum_{Q\subseteq N}\mathcal{I}(Q) = \Delta(N)$.
Aksiom 2: Vzemimo i-ti in j-ti atribut ($i \neq j$). Če pogledamo enačbo za prispevek atributa (glej enačbo (3.4)), vidimo, da členi $\mathcal{I}(S), S \subseteq N \setminus \{i,j\}$ ne nastopajo ne pri φ_i ne pri φ_j, členi $\mathcal{I}(S\cup\{i,j\}), S \subseteq N \setminus \{i,j\}$ pa nastopajo pri obeh. Prispevka φ_i in φ_j se razlikujeta le v členih z $\mathcal{I}(S\cup i)$ oziroma $\mathcal{I}(S\cup j)$, kjer je $S \subseteq N \setminus \{i,j\}$. Dovolj bo, če pokažemo, da so ti členi paroma enaki. Z uporabo pogoja pri 2. aksomu v razvoju enačbe (3.3) za $\mathcal{I}(S\cup\{i\}), S \subseteq N \setminus \{i,j\}$ zamenjamo člene $\Delta(W\cup\{i\})$ z $\Delta(W\cup\{j\})$ (členov, kjer nastopata oba, ni). Sledi $\mathcal{I}(S\cup i) = \mathcal{I}(S\cup\{j\})$, za vsak $S \subseteq N \setminus \{i,j\}$ in $\varphi_i = \varphi_j$.
Aksiom 3: Naj za i-ti atribut velja $\Delta(S\cup\{i\}) = \Delta(S)$, za vsak $S \subset N \setminus \{i\}$. Neposredno iz tega sledi $\mathcal{I}(\{i\}) = 0$. Z uporabo slednjega pokažemo, da za poljuben $j \neq i$ velja $\mathcal{I}(\{j,i\}) = 0$. Nadaljujemo z indukcijo po številu elementov in pokažemo za poljubno velik N, da za vsak $Q \subset N \setminus \{i\}$ velja $\mathcal{I}(Q\cup\{i\}) = 0$. Iz te ugotovitve in enačbe (3.4) sledi $\varphi_i = 0$.
Aksiom 4: Dokazati moramo, da je vsota prispevkov dveh iger $\Delta_1(S)$ in $\Delta_2(S)$ enaka prispevku pri igri vsote. Torej, $\varphi_i(\Delta_1+\Delta_2) = \varphi_i(\Delta_1) + \varphi_i(\Delta_2)$, kjer pri igri vsote velja $(\Delta_1+\Delta_2)(S) = \Delta_1(S)+\Delta_2(S)$. Z uporabo te enakosti na vsoti desnih strani enačbe (3.3) za $\mathcal{I}_1(S)$ in $\mathcal{I}_2(S)$ pokažemo, da velja $(\mathcal{I}_1+\mathcal{I}_2)(S) = \mathcal{I}_1(S)+\mathcal{I}_2(S)$, kjer je $(\mathcal{I}_1+\mathcal{I}_2)(S)$. S tem je dokaz zaključen. □

Dokaz 2. Ta dokaz smo uporabili v predhodnih raziskavah [96, 98], ko smo rekurzivno definicijo prispevka (3.4) preobrazili v eksplicitno obliko. Najprej rekurzivno definicijo interakcij $\mathcal{I}$ (glej enačbo 3.3) preobrazimo v eksplicitno obliko

$$\mathcal{I}(S) = \sum_{W\subseteq S}((-1)^{|S|-|W|}\Delta(W)). \tag{3.5}$$

Izpeljavo (3.5) dokažemo z indukcijo. Enačbi (3.4) in (3.5) združimo v eksplicitno enačbo za prispevek atributa:

$$\varphi_i(\Delta) = \sum_{W\subseteq N\setminus\{i\}} \frac{\sum_{Q\subseteq(W\cup\{i\})}((-1)^{|W\cup\{i\}|-|Q|}\Delta(Q))}{|W\cup\{i\}|}. \tag{3.6}$$

Sedaj preštejmo kolikokrat se $\Delta(S\cup\{i\}), S \subseteq N, i \notin S$ pojavi na desni strani enačbe (3.6). Naj bo $M_{\Delta(S\cup\{i\})}$ število vseh takih pojavitev in $k = n-s-1$, $s = |S|$. Izraz $\Delta(S\cup\{i\})$ se pojavi, ko je $S \subseteq W$ in samo enkrat za vsak W. Za W, kjer je $S \subseteq W$ in $|W| = s+l$, $w = |W|$, se $\Delta(S\cup\{i\})$ pojavi z alternirajočim prezdnakom, odvisno od parnosti a. Obstaja natanko $\binom{k}{l}$ takih W v enačbi (3.6), saj lahko uporabimo poljubno kombinacijo l dodatnih elementov izmed k elementov, ki niso že v S.

Ko zapišemo vse take izraze do vključno $W = N$ in upoštevamo, da vsako interakcijo $\mathcal{I}(W)$ delimo z w, dobimo zaporedje:

$$M_{\Delta(S\cup\{i\})} = \frac{\binom{k}{0}}{n-k} - \frac{\binom{k}{1}}{n-k+1} + ... + (-1)^k \frac{\binom{k}{k}}{n} = V(n,k)$$

Na podoben način obravnavamo $\Delta(S)$, $i \notin S$, kjer dobimo $M_{\Delta(S)} = -V(n,k)$. Zaporedje $V(n,k)$ lahko izrazimo z beta funkcijo:

$$(1-x)^k = \binom{k}{0} - \binom{k}{1}x + \binom{k}{2}x^2 - ... \pm \binom{k}{k}x^k$$

$$\int_0^1 x^{n-k-1}(1-x)^k\,dx = \int_0^1 (\binom{k}{0}x^{n-k-1} - \binom{k}{1}x^{n-k} + ... \pm \binom{k}{n-1}x^{n-1})\,dx$$

$$B(n-k,k+1) = \frac{\binom{k}{0}}{n-k} - \frac{\binom{k}{1}}{n-k+1} + ... \pm \frac{\binom{k}{k}}{n} = V(n,k).$$

Uporabimo $B(p,q) = \frac{\Gamma(p)\Gamma(q)}{\Gamma(p+q)}$ in dobimo $V(n,k) = \frac{(n-k-1)!k!}{n!}$. Sledi:

$$\varphi_i(\Delta) = \sum_{S\subseteq N\setminus\{i\}} V(n,n-s-1)\cdot\Delta(S\cup\{i\}) - \sum_{S\subseteq N\setminus\{i\}} V(n,n-s-1)\cdot\Delta(S) =$$
$$= \sum_{S\subseteq N\setminus\{i\}} \frac{(n-s-1)!s!}{n!}\cdot(\Delta(S\cup\{i\}) - \Delta(S)).$$

S tem je dokaz zaključen.

□

Enačbo (3.6) zapišemo tudi kot

$$\varphi_i = \sum_{S\subseteq N\setminus\{i\}} \frac{(n-s-1)!s!}{n!}\cdot(\Delta(S\cup\{i\}) - \Delta(S)). \tag{3.7}$$

Iz izreka 3.3 sledi, da prispevki, ki jih izračunamo s predlagano metodo, izpolnjujejo vse štiri Shapleyeve aksiome. Prvi trije so v konktekstu prispevkov atributov k napovedi še posebej uporabni.

Prvi aksiom pravi, da bo vsota prispevkov enaka razliki med pričakovano napovedjo in dejansko napovedjo za primer $\vec{a}$. Prispevki so implicitno normalizirani in omejeni z razliko v napovedi. Drugi aksiom pravi, da bosta atributa, ki imata identičen učinek na vse podmnožice preostalih atributov, dobila enak prispevek. Po tretjem aksiomu bo atributu, ki nima prav nobenega vpliva na napoved, dodeljen ničelni prispevek.

Prispevki, ki jih izračunamo po predlagani metodi, imajo te želene lastnosti. Poleg tega je metoda edini način kako razdeliti razliko v napovedi $\Delta(N)$ med atribute, ki ima vse štiri lastnosti.

3.2.2 Ilustrativna primera

S primeroma **diskunkcije** in **ekskluzivnega-ali** smo izpostavili pomanjkljivosti obstoječih univerzalnih metod za računanje prispevkov atributov. Pokažimo, da je predlagana metoda bolj uspešna. Obenem sta primera v pomoč pri razumevanju delovanja metode. Uporabimo enačbe (3.4) (3.5), in (3.6), ki so v primerjavi z neposrednim izračunom (3.7) bolj ilustrativne.

Začnimo s primerom **ekskluzivnega-ali**. Naj bo $\mathcal{X} = \{0,1\} \times \{0,1\}$ in $f(x_1, x_2) = x_1$ xor x_2. Zaradi lažjega računanja vzemimo kontekst, v katerem so vsi primeri enako verjetni. Izračunajmo prispevke za napoved $f(1,0) = 1$.

Pričakovane napovedi so $E[f] = E[f|x_1 = 1] = E[f|x_2 = 0] = \frac{1}{2}$ in $E[f|x_1 = 1 \wedge x_2 = 0] = 1$. Sledi $\Delta(\{1\}) = \Delta(\{2\}) = 0$ in $\Delta(\{1,2\}) = \frac{1}{2}$. Vidimo, da pri xor šele oba atributa skupaj povzročita razliko v napovedi.

Sedaj lahko izračunamo interakcije. Najprej $\mathcal{I}(\{1\}) = \mathcal{I}(\{2\}) = 0$, nato še $\mathcal{I}(\{1,2\}) = \frac{1}{2}$. Celotna razlika v napovedi izvira iz interakcije med atributoma, ki sama zase nič ne prispevata. Preostane nam, da edino neničelno interakcijo razdelimo med oba atributa. Dobimo $\varphi_1 = \varphi_2 = \frac{1}{4}$.

Nadaljujmo z drugim ilustrativnim primerom - **disjunkcijo**. Tokrat računanje dodatno zapletemo z uporabo treh atributov $\mathcal{X} = \{0,1\} \times \{0,1\} \times \{0,1\}$. Model naj bo $f(x_1, x_2, x_3) = x_1 \vee x_2 \vee x_3$. Spet uporabimo kontekst, kjer so vsi primeri enako verjetni, in izračunajmo prispevke k napovedi $f(1,1,0) = 1$.

Ponovno začnemo pri pogojnih napovedih. Z izjemo $E[f] = \frac{7}{8}$ in $E[f|x_3 = 0] = \frac{6}{8}$, so vse preostale relevantne pogojne napovedi enake 1, saj je že ena enica dovolj, da napovemo 1. Členi Δ so $\Delta(\{3\}) = -\frac{1}{8}$ in $\Delta(\{1\}) = \Delta(\{2\}) = \Delta(\{1,2\}) = \Delta(\{1,3\}) = \Delta(\{2,3\}) = \Delta(\{1,2,3\}) = +\frac{1}{8}$.

Interakcije atributov samih s seboj so tokrat neničelne $\mathcal{I}(\{1\}) = \mathcal{I}(\{2\}) = \frac{1}{8}$ in $\mathcal{I}(\{3\}) = -\frac{1}{8}$, saj atributi že sami zase prispevajo k napovedi. Vrednosti interakcij parov so $\mathcal{I}(\{1,3\}) = \mathcal{I}(\{2,3\}) = \frac{1}{8}$ in $\mathcal{I}(\{1,2\}) = -\frac{1}{8}$. Tukaj srečamo prvo negativno interakcijo dveh ali več atributov. To je razumljivo, saj prva dva atributa skupaj prispevata manj, kot bi pričakovali iz njunih posameznih prispevkov. Pričakovano je tudi interakcija vseh treh $\mathcal{I}(\{1,2,3\}) = \Delta(\{1,2,3\}) - \sum_{Q \subset \{1,2,3\}} \mathcal{I}(Q) = -\frac{1}{8}$ negativna.

Interakcije še razdelimo med udeležene atribute. Prispevek tretjega atributa je $\varphi_3 = \mathcal{I}(\{3\}) + \frac{\mathcal{I}(\{1,3\})+\mathcal{I}(\{2,3\})}{2} + \frac{\mathcal{I}(\{1,2,3\})}{3} = -\frac{1}{24}$. Prispevka prvih dveh atributov pa sta zaradi simetričnega vpliva teh atributov enaka, zato izračunamo le prvega $\varphi_1 = \mathcal{I}(\{1\}) + \frac{\mathcal{I}(\{1,2\})+\mathcal{I}(\{1,3\})}{2} + \frac{\mathcal{I}(\{1,2,3\})}{3} = \frac{1}{12} = \varphi_2$. Prispevki so v skladu z našo intuicijo, da enice prispevajo k napovedi 1, ničla pa prispeva v nasprotno smer.

3.3 Aproksimacija prispevka atributa

Enačba 3.7 je teoretično dobro utemeljena in ima želene lastnosti, vendar je v praksi pogosto nemogoče izračunati člene Δ. Četudi bi te člene uspeli izračunati v sprejemljivem

času, ima izračun φ_i še vedno eksponentno časovno zahtevnost. Za praktično rabo moramo zmanjšati časovno zahtevnost, kar storimo s pomočjo aproksimacije.

Najprej zapišimo enačbo (3.7) v njeni ekvivalentni obliki

$$\varphi_i(\Delta) = \frac{1}{n!} \sum_{\mathcal{O}\in\pi(N)} \left(\Delta(Pre^i(\mathcal{O}) \cup \{i\}) - \Delta(Pre^i(\mathcal{O}))\right), \quad i = 1, ..., n, \tag{3.8}$$

kjer je $\pi(N)$ množica vseh urejenih permutacij množice N in $Pre^i(\mathcal{O})$ množica indeksov vseh atributov, ki so predhodniki atributa i v vrstnem redu $\mathcal{O} \in \pi(N)$.

Dokaz ekvivalence (3.7) in (3.8). Preštejmo kolikokrat se neka določena podmnožica S pojavi kot $\Delta(S \cup \{i\})$ na desni strani enačbe (3.8). Natanko tolikokrat, kolikor je permutacij n atributov, pri katerih so atributi iz S na prvih s mestih, nato atribut i, na zadnjih $n - s - 1$ mestih pa so preostali atributi $N \setminus (S \cup \{i\})$. Takih permutacij je $s!(n - s - 1)!$ in, če upoštevamo še deljenje z $n!$ izven vsote, je ekvivalenca jasna. □

Z uporabo primera bančnih roparjev si enačbo (3.8) interpretiramo na sledeč način. Roparje najprej postavimo vrsto (v poljubnem vrstnem redu), nato prvemu v vrsti dodelimo del plena enak razliki med naropano vsoto in vsoto, ki bi jo naropali, če bi ropali le preostali roparji. Ta ropar nato odide, mi pa ponavljamo postopek, dokler ne pridemo do (vključno) zadnjega roparja. Končna Shapleyeva vrednost za posameznega roparja je povprečna vsota, ki bi jo ropar dobil, če postopek ponovimo za vse možne vrstne rede.

Za izračun Shapleyeve vrednosti že obstaja ustrezen aproksimacijski algoritem (glej [16]), vendar se v našem primeru v enačbi (3.8) eksponentna časovna zahtevnost skriva tudi v izračunu členov Δ. Najprej predpostavimo, da je izbrani kontekst p tak, da so posamezni atributi popolnoma neodvisni, če jih obravnavamo kot slučajne spremenljivke v verjetnostnem prostoru $(\mathcal{X}, \mathcal{P}(\mathcal{X}), p')$. Pri tem je potrebno poudariti, da se predpostavka nanaša na neodvisnost vhodnih spremenljivk in ne na morebitno neodvisnost njihovega vpliva na napoved - še vedno upoštevamo vse potencialne interakcije.

Od tu dalje večkrat uporabimo notacijo $\vec{x}_{[...]}$ za vektorje, ki jim spremenimo določene vrednosti v skladu z navodili [...]. Z $\vec{x}_{[x_2=X_{2,j}]}$ bi tako zapisali vektor $\vec{x}$, ki mu drugo komponento nastavimo na j-to vrednost drugega atributa. Ali pa $\vec{x}_{[x_i=y_i, i\in S]}$, vektor $\vec{x}$, katerega komponente na mestih v S nastavimo na vrednosti vektorja $\vec{y}$. Pri $\vec{x} = \langle 2, 4, 6\rangle$ in $\vec{y} = \langle 3, 5, 7\rangle$ je tako $\vec{x}_{[x_i=y_i, i\in\{1,3\}]} = \langle 3, 4, 7\rangle$.

Z uporabo predpostavke preobrazimo enačbo (3.1) v

$$\begin{aligned} \Delta(S) &= E[f|S] - E[f|\varnothing] = \\ &= \sum_{\vec{x}\in\mathcal{X};\forall i: x_i = a_i \vee i \notin S} p'(\vec{x}) f(\vec{x}) - \sum_{\vec{x}\in\mathcal{X};\forall i: x_i = a_i} p'(\vec{x}) f(\vec{x}) = \\ &= \sum_{\vec{x}\in\mathcal{X}} p'(\vec{x})(f(\vec{x}_{[x_i=a_i, i\in S]}) - f(\vec{x}) \end{aligned} \tag{3.9}$$

Sedaj lahko v enačbi (3.8) zamenjamo člene Δ z izrazom iz zgornje enačbe (3.9) in dobimo

$$\varphi_i = \frac{1}{n!} \sum_{\mathcal{O} \in \pi(N)} \sum_{\vec{x} \in \mathcal{X}} p(\vec{x}) \cdot (f(\vec{x}_{[x_j = a_j, j \in Pre^i(\mathcal{O}) \cup \{i\}]}) - f(\vec{x}_{[x_j = a_j, j \in Pre^i(\mathcal{O})]})). \tag{3.10}$$

Sedaj uporabimo sledeč način vzorčenja. Naša populacija je $\pi(N) \times \mathcal{X}$ in vsak par $\langle$ vrstni red / primer $\rangle$ predstavlja en vzorec $V_{\mathcal{O},\vec{x} \in \mathcal{A}} = f(\tau(\vec{a}, \vec{x}, Pre^i(\mathcal{O}) \cup \{i\})) - f(\tau(\vec{a}, \vec{x}, Pre^i(\mathcal{O})))$. Če vzorčimo naključno in z vračanjem, je verjetnost, da izvlečemo vzorec $V_{\mathcal{O},\vec{x} \in \mathcal{A}}$ enaka $p'(\vec{x})$. Sedaj na ta način izvlecimo m vzorcev $V_1, V_2, ..., V_m$ in definirajmo slučajno spremenljivko

$$\hat{\varphi}_i = \frac{1}{m} \sum_{j=1}^{m} V_j. \tag{3.11}$$

Iz centralnega limitnega izreka sledi, da je $\hat{\varphi}_i$ približno normalno porazdeljena slučajna spremenljivka, katere upanje je φ_i, varianca pa $\frac{\sigma_i^2}{m}$, kjer je σ_i^2 varianca populacije. Cenilka $\hat{\varphi}_i$ je nepristrana in dosledna cenilka prispevka φ_i.

3.3.1 Aproksimacijski algoritem in napaka aproksimacije

Enačba (3.11) opisuje nepristrano in dosledno cenilko, ki jo lahko izračunamo na podlagi m vzorcev. V praksi nas zanima predvsem, kako velik mora biti m, da bo napaka aproksimacije dovolj majhna.

Za vsak φ_i je napaka vzorčenja odvisna le od variance populacije σ_i^2. Čeprav je v praksi σ_i^2 neznana, lahko določimo zgornjo mejo. Ob normalizirani odvisni spremenljivki (pri klasifikaciji so napovedi že implicitno normalizirane, saj gre za verjetnost) dosežemo maksimalno varianco, če je populacija razdeljena med obe ekstremni vrednosti 1 in -1, ko je $\sigma_i^2 \leq 1$.

S parom $\langle 1 - \alpha, e \rangle$ bomo opisali željo, da napaka izpolnjuje sledečo omejitev $P(|\varphi_i - \hat{\varphi}_i| < e) = 1 - \alpha$. Z besedami, želimo, da ima prispevek manjšo napako od e z verjetnostjo $1 - \alpha$. Za določen par $\langle 1 - \alpha, e \rangle$ potrebujemo končno mnogo vzorcev, da zadostimo pogoju. To število je

$$m_i(\langle 1 - \alpha, e \rangle) = \frac{Z_{1-\alpha}^2 \cdot \sigma_i^2}{e^2}, \tag{3.12}$$

kjer je $Z_{1-\alpha}^2$ Z-ocena za $1 - \alpha$ interval zaupanja. Enačbo (3.12) lahko uporabimo tudi v nasprotni smeri, za izračun poljubnega intervala zaupanja za aproksimacijo pri danem številu vzorcev m_i.

Poglejmo si še na primeru. Recimo, da je naša želja, da je 99% izračunanih prispevkov za manj kot 0.01 oddaljeno od dejanskih vrednosti. Ob predpostavki, da je varianca najvišja možna $\sigma_i^2 = 1, \forall i \in N$, bomo potrebovali približno 65000 vzorcev, ne glede na število atributov n. Kot pokažemo v 4. poglavju, so v praksi variance nižje.

Algoritem 1 Algoritem za aproksimacijo prispevka i-tega atributa k napovedi modela f za primer $\vec{a}$. Vzorčimo m vzorcev.

$\varphi_i(\vec{a}) \leftarrow 0$
for 1 to m **do**
naključno izberemo permutacijo $\mathcal{O} \in \pi(n)$
naključno izberemo $\vec{x} \in \mathcal{X}$
sestavimo nova primera:

$$\vec{b}_1 \leftarrow \overbrace{\boxed{\text{predhodniki } i\text{-tega v } \mathcal{O}}\ \boxed{i}}^{\text{vrednosti vzamemo iz } \vec{a}}\ \overbrace{\boxed{\text{nasledniki } i\text{-tega v } \mathcal{O}}}^{\text{vrednosti vzamemo iz } \vec{x}}$$

$$\vec{b}_2 \leftarrow \overbrace{\boxed{\text{predhodniki } i\text{-tega v } \mathcal{O}}}^{\text{vrednosti vzamemo iz } \vec{a}}\ \overbrace{\boxed{i}\ \boxed{\text{nasledniki } i\text{-tega v } \mathcal{O}}}^{\text{vrednosti vzamemo iz } \vec{x}}$$

$\varphi_i(\vec{a}) \leftarrow \varphi_i(\vec{a}) + f(\vec{b}_1) - f(\vec{b}_2)$

end for
$\varphi_i(\vec{a}) \leftarrow \frac{\varphi_i(\vec{a})}{m}$

Za izračun prispevka atributa k napovedi za primer $\vec{a}$ predlagamo Algoritem 1, ki je neposredna implementacija izračuna enačbe (3.11). Časovna zahtevnost algoritma je odvisna tudi od časovne zahtevnosti napovedi modela, a jo lahko približno določimo. Pri razlagi za nek primer moramo algoritem izvesti n-krat, za vsak atribut po enkrat. Pri dani napaki je časovna zahtevnost $c \cdot O(n \cdot T(f, \mathcal{X}))$, kjer je $T(f, \mathcal{X})$ časovna zahtevnost napovedi modela f.

V praksi za večino klasifikacijskih in regresijskih modelov velja $T(f, \mathcal{X}) \leq n$. Konstanta c označuje število vzorcev, ki jih potrebujemo, da dosežemo želeno omejitev napake, in je neodvisna od števila atributov n. Časovna zahtevnost Algoritma 1 je v večini primerov kvadratna.

3.3.2 Postopek vzorčenja za enake pričakovane napake

V prejšnjem podpoglavju smo predstavili aproksimacijski algoritem za prispevek i-tega atributa φ_i, ki nam omogoča, da razmerje med časom računanja in napako aproksimacije uravnavamo z izbiro števila vzorcev m. Večkrat ko vzorčimo, manjša bo napaka in več časa bomo potrebovali za računanje. Ponudili smo tudi enačbo za število vzorcev, ki nam pri varianci populacije σ_i^2 zagotavlja, da bo napaka znotraj e z verjetnostjo $1 - \alpha$ (glej enačbo (3.12)).

To je vse, kar v teoriji potrebujemo za izbiro ustreznega števila vzorcev za vsak atribut. V praksi pa se stvari nekoliko zapletejo. Iz enačbe (3.12)) je takoj razvidno, da je v praksi za dosego enake omejitve napake za vse atribute potrebno število vzorcev, ki ga vzorčimo

za atribut i, prilagoditi varianci σ_i^2. Izbira enakega števila vzorcev za vse atribute ni vedno optimalna.

Ker varianca ni vnaprej znana, jo lahko ocenimo šele, ko že vzorčimo nekaj vzorcev za i-ti atribut. V praksi bi vzeli nekaj vzorcev za vsak atribut, ocenili varianco in izračunali število vzorcev, ki jih potrebujemo. Težava je v tem, da ob ponovnem vzorčenju dobimo boljšo oceno variance kot je bila začetna, zato je število vzorcev smiselno prilagajati kar sproti.

Tu se pojavi novo vprašanje, kako pogosto ponovno oceniti varianco. Ob vsem tem smo predpostavili, da uporabnik ni omejen s časom. V slednjem primeru bi bilo potrebno čas računanja ustrezno razporediti med atribute, da bi na koncu vsi imeli enako pričakovano napako. Pri tem se nam seveda ne sme zgoditi, da nam za nek atribut zmanjka časa, zato je smiselno atribute obravnavati vzporedno in ne enega za drugim. Očitno je za bolj preprosto rabo metode za razlago potrebna sistematična rešitev, zato predlagamo razširitev Algoritma 1 za samodejno izbiro števila vzorcev $m_i, i \in N$.

Za lažje razumevanje ideje za algoritem si predstavljamo sledečo situacijo. Po nekem postopku jemljemo vzorce za prispevke atributov v primeru $\vec{a}$. Do tega trenutka smo vzeli že $\sum_n m_i'$ vzorcev, kjer je $m_i' > 1$ število vzorcev, ki smo jih do tega trenutka vzeli za i-ti atribut. Naj bodo $\mathcal{V}_i = V_{1,i}, V_{2,i}, ..., V_{m_i',i}$ vzorci za i-ti atribut. Če bi v tem trenutku prekinili vzorčenje, bi si želeli, da je razmerje $\frac{m_i'}{\sigma_i^2}$ enako za vse atribute $i \in N$. Natančneje $\frac{m_i'}{\sigma_i^2} = \frac{\sum_n m_i'}{\sum_n \sigma_i^2}$ oziroma $\frac{m_i'}{\sum_n m_i'} = \frac{\sigma_i^2}{\sum_n \sigma_i^2}$. To bi namreč pomenilo, da je smo za vse atribute enako omejili napako (glej enačbo (3.12)). Pri tem smo varianco za i-ti atribut ocenili iz $\mathcal{V}_i$.

Želimo algoritem, ki teži k minimizaciji vrednosti izraza $\mathcal{E} = \sum_n |\frac{m_i'}{\sum_n m_i'} - \frac{\sigma_i^2}{\sum_n \sigma_i^2}|$. V naslednjem koraku lahko za vsak atribut preverimo, kako bi se spremenila vrednost $\mathcal{E}$, če bi vzeli vzorec za ta atribut (predpostavimo, da se σ_i^2 ne spremeni). Nato vzorčimo za atributa, ki minimizira $\mathcal{E}$ v naslednjem koraku. Nekoliko bolj enostaven način za dosego tega, da bodo deleži vzorcev na dolgi rok enak deležem varianc, je izbira naslednjega atributa v skladu z deleži varianc. Atribut, katerega vzorec bomo vzeli v naslednjem koraku, izberemo naključno, pri čemer je i-ti atribut izbran z verjetnostjo $\frac{\sigma_i^2}{\sum_n \sigma_i^2}$.

Algoritem 2 Algoritem inkrementalno računanje variance. Naj bo k trenutni korak, μ trenutna aritmetična sredina in $M2$ trenutna vsota kvadratov razlike med členi in aritmetično sredino. Vrednost naslednjega člena je x.

$k \leftarrow k + 1$
$d \leftarrow x - \mu$
$\mu \leftarrow \mu + \frac{d}{k}$
$M2 \leftarrow M2 + d(x - \mu)$
$\sigma^2 \leftarrow \frac{M2}{k-1}$

Ključen del algoritma je tudi računanje variance, ki ga moramo ponoviti za vsak vzorec. Uporabimo Knuthov predlog inkrementalnega algoritma za ocenjevanje variance

populacije, ki temelji na ugotovitvah Welforda [47, 99] (glej Algoritem 2). Algoritma 1 in 2 združimo v Algoritem 3 za računanje prispevkov vseh atributov hkrati, z optimizirano izbiro vzorcev na podlagi ocenjenih varianc. Pogoj za ustavitev izvajanja lahko sestavimo iz ene ali več omejitev. V tem delu uporabljamo omejitev skupnega števila vzorcev, možne pa so tudi druge vrste omejitev (časovna, ko dosežemo želeno omejitev napake pri vseh atributih, ipd...).

Poleg pogoja za ustavitev ima algoritem en sam parameter $m_{min} > 1$, ki določa, koliko vzorcev vzorčimo za vsak atribut na začetku, da dobimo približno oceno variance.

Algoritem 3 Algoritem za aproksimacijo prispevkov vseh atributov k napovedi modela f za primer $\vec{a}$. Z m_{min} označimo začetno število vzorcev, ki jih vzorčimo za vsak atribut. Pogoj za prekinitev je lahko vezan na skupno število vzorcev ali čas izvajanja.

---

for $i = 1$ to n **do**
  $k_i \leftarrow 0$, $\mu_i \leftarrow 0$, $M2_i \leftarrow 0$, $\varphi_i(\vec{a}) \leftarrow 0$
end for
while pogoj za prekinitev ni izpolnjen ali $\exists i : k_i < m_{min}$ **do**
  if $\forall i : k_i \geq m_{min}$ **then**
    naključno izberemo j-ti atribut, pri čemer je i-ti atribut izbran z verjetnostjo $\frac{\sigma_i^2}{\sum_n \sigma_i^2}$ †
  else
    izberemo najmanši tak j, da je $k_j < m_{min}$
  end if
  temp$\varphi \leftarrow$ rezultat Algoritma 1 za en vzorec in j-ti atribut
  $\sigma_i^2 \leftarrow$ rezultat Algoritma 2 za tempφ, k_j, μ_j in $M2_j$
  $\varphi_j(\vec{a}) \leftarrow \varphi_j(\vec{a})$+temp$\varphi$
end while
for $i = 1$ to n **do**
  $\varphi_i(\vec{a}) \leftarrow \frac{\varphi_i(\vec{a})}{k_i}$
end for

† Če želimo minimizirati pričakovano absolutno napako, izberemo tisti atribut j, pri katerem je vrednost izraza $\sqrt{\frac{\sigma_i^2}{k_i}} - \sqrt{\frac{\sigma_i^2}{k_i+1}}$ maksimalna.

3.3.3 Postopek vzorčenja za minimalno pričakovano skupno napako

Pogosto ne želimo, da imajo vsi prispevki enako pričakovano napako, temveč želimo minimizirati skupno napako aproksimacije. Čeprav sprva morda ne opazimo razlike, lahko pokažemo, da slednje dosežemo z razporeditvijo vzorcev, ki ni nujno sorazmerna varianam atributov.

Kot smo že ugotovili, je cenilka prispevka $\hat{\varphi}_i \approx N(\varphi_i, \frac{\sigma_i^2}{m_i})$ približno normalno porazdeljena. Sledi $\hat{\varphi}_i - \varphi_i \approx N(0, \frac{\sigma_i^2}{m_i})$. Porazdelitev absolutne napake aproksimacije i-tega prispevka $Y_i \approx |\hat{\varphi}_i - \varphi_i|$ je polovica normalne porazdelitve $\hat{\varphi}_i - \varphi_i \approx N(0, \frac{\sigma_i^2}{m_i})$. Matematično upanje slednje je

$$E[Y_i] = \sigma\sqrt{\frac{2}{\pi}},$$

kjer je σ varianca normalne porazdelitve, iz katere izhajamo. V našem primeru

$$E[Y_i] = \sqrt{\frac{\sigma_i^2}{m_i}}\sqrt{\frac{2}{\pi}}.$$

Pričakovana vsota napak je enaka

$$E[\sum_{i=1}^{n} tY_i] = \sum_{i=1} n\sqrt{\frac{\sigma_i^2}{m_i}}\sqrt{\frac{2}{\pi}} = \sqrt{\frac{2}{\pi}}\sum_{i=1}^{n}\frac{\sigma_i}{\sqrt{m_i}}.$$

Da bi minimizirali pričakovano napako, moramo skupno vsoto vzorcev m med m_i razporediti tako, da minimiziramo vrednost izraza $\sum_{i=1}^{n}\frac{\sigma_i}{\sqrt{m_i}}$. Gre za optimizacijski problem. Pokažimo, da lahko uporabimo kar pozrešen algoritem.

Naj bodo znane variance $\sigma_i \geq 0$ in $m = \sum_{i=0}^{n} m_i$ optimalna rešitev v m-tem koraku. V $m+1$-tem koraku bo rešitev $m+1 = \sum_{i=0}^{n} m'_i$ za $m'_i = m_i, \forall i \neq j$ in $m'_j = m_j + 1$, kjer je

$$j \in \operatorname*{argmax}_i \frac{\sigma_i}{\sqrt{m_i}} - \frac{\sigma_i}{\sqrt{m_i+1}},$$

tudi optimalna. Z drugimi besedami, dovolj je, da v vsakem koraku vzorčimo vzorec za atribut, pri katerem s tem najbolj zmanjšamo vrednost vsote $\sum_{i=1}^{n}\frac{\sigma_i}{\sqrt{m_i}}$.

Dokaz. Upoštevamo, da je funkcija $f(x) = \frac{\sigma_i}{\sqrt{x}}$, pri čemer je $\sigma_i \geq 0$, padajoča na $\Re^+$ (strogo padajoča za $\sigma_i > 0$!). Izbira, ki v danem trenutku najbolj zmanjša vrednost kriterijske funkcije, jo zmanjša tudi bolj od vseh možnih izbir v prihodnosti. Sledi, da je vsaka rešitev, ki od tega koraka naprej ne vsebuje te izbire, slabša, kvečjemu enako dobra.

Pozor! Če predpostavimo, da je $\sigma_i > 0, \forall i$, potem nam stroga monotonost zgoraj opisane funkcije zagotavlja, da za opisani postopek obstaja tak $m' > m$, pri katerem pridemo do razporeditve, ki je za m' optimalna, četudi začetna razporeditev za m ni optimalna. □

Opisani postopek nam zagotavlja, da iz optimalne rešitve za m preidemo v optimalno rešitev za $m+1$. Nismo pa še pojasnili, kako dobimo začetno optimalno rešitev. Če bi bile σ_i^2 vnaprej znane in pozitivne, potem za optimalno začetno rešitev izberemo $m_i = 1, \forall i$.

V praksi variance niso vnaprej znane, zato smo prisiljeni najprej vzeti nekaj vzorcev za vsak atribut, da ocenimo σ_i^2. Predpostavimo, da za vsak atribut vzorčimo enako število

začetnih vzorcev $m_{min} \geq 2$. Za vsak atribut i, za katerega velja $\sigma_i = 0$, smo vzeli m_{min} vzorcev, od katerih nimamo nobene koristi. V najslabšem primeru, ko je $\sigma_i = 0, \forall i$, smo brez potrebe vzeli $n \cdot m_{min}$ vzorcev. Za vse $i, \sigma_i^2 > 0$ so vzeti vzorci koristni in imamo zagotovilo, da bomo z zgoraj opisanim postopkom iz začetne rešitve $m_i = m_{min}, \forall i$ prišli do nekega $m' > m$, pri katerem bo razporeditev optimalna (glej prejšnji dokaz). Ni pa nujno, da bomo tako razporeditev v praksi tudi dosegli - število vzorcev, ki jih vzorčimo, mora biti večje od m'.

V praksi bi si želeli vzeti čim manjši m_{min}, da zmanjšamo število nepotrebnih vzorcev. Po drugi strani ne želimo napačno oceniti variance in atribut označiti za nepomembnega. Z izbiro minimalnega števila vzorcev za vsak atribut v praksi tehtamo med verjetnostjo, da bomo pomembnemu atributu dodelili prispevek 0, in številom vzorcev, ki jih bomo potencialno zavrgli.

Algoritem za razporejanje vzorcev z namenom minimalne pričakovane skupne napake je zajet v algoritmu 3. Korak, označen z †, zamenjamo s tistim v opombi. Na koncu je potrebno omeniti, da bolj učinkovita razporeditev vzorcev bolj smotrno izrabi čas, ki je na voljo za računanje, a ne vpliva na časovno zahtevnost algoritma. Pridobitek na račun optimalne izbire vzorcev smo v naslednjem poglavju izmerili z uporabo nabora različnih modelov in množic podatkov.

3.4 Splošen prispevek vrednosti atributa in njegova pomembnost

Metoda, ki smo jo predlagali, računa prispevek atributa k napovedi za določen primer - nivo φ. Že v uvodu smo omenili, da se, ko imamo na voljo prispevke atributov za neko množico primerov, vprašamo tudi, kako te prispevke združiti. Najprej v splošne prispevke vrednosti atributov (ϕ-nivo), ki nam dajo nekoliko širšo sliko o vplivu posameznih vrednosti. Na koncu pa v najvišji nivo - pomembnost atributa (Λ-nivo), ki nam atribute pomaga urediti po pomembnosti za model.

Predstavljamo si, da smo izračunali prispevek φ_i, i-tega atributa, za vse take primere $\vec{x}$, kjer ima i-ti atribut j-to vrednost. Sedaj te prispevke povprečimo, da dobimo povprečen prispevek j-te vrednosti i-tega atributa. Časovni zahtevnosti takega početja se izognemo na sledeč način.

Definirajmo splošen prispevek j-te vrednosti i-tega atributa $\phi_{i,j}$, kot uteženo povprečje prispevka i-tega atributa φ_i (glej enačbi (3.8) in (3.10)) preko vseh primerov, kjer je

njegova vrednost j-ti element množice vrednosti atributa:

$$\begin{aligned}
\psi_{i,j} = &\sum_{\vec{x}\in\mathcal{X}, x_i=\mathcal{X}_{i,j}} p'(\vec{x}|x_i=\mathcal{X}_{i,j})\varphi_i(\vec{x}) = \sum_{\vec{x}\in\mathcal{X}} p'(\vec{x})\varphi_i(\vec{x}') = \\
=&\sum_{\vec{x}\in\mathcal{X}} p'(\vec{x}) \left(\frac{1}{n!} \sum_{\mathcal{O}\in\pi(N)} \sum_{\vec{y}\in\mathcal{X}} p'(\vec{y}) (f(\vec{y}_{[y_j=x_j, j\in Pre^i(\mathcal{O})\cup\{i\}]}) - f(\vec{y}_{[y_j=x_j, j\in Pre^i(\mathcal{O})]})) \right) = \\
=&\frac{1}{n!} \sum_{\mathcal{O}\in\pi(N)} \left(\sum_{\vec{x}\in\mathcal{X}} p'(\vec{x}) \sum_{\vec{y}\in\mathcal{X}} (f(\vec{y}_{[y_j=x_j, j\in Pre^i(\mathcal{O})\cup\{i\}]}) - f(\vec{y}_{[y_j=x_j, j\in Pre^i(\mathcal{O})]})) \right) = \\
=&\frac{1}{n!} \sum_{\mathcal{O}\in\pi(N)} \sum_{\vec{z}\in\mathcal{X}} p'(\vec{z}) \left(f(\vec{z}') - f(\vec{z})\right) \\
=&\sum_{\vec{z}\in\mathcal{X}} p'(\vec{z}) \left(f(\vec{z}') - f(\vec{z})\right),
\end{aligned} \tag{3.13}$$

kjer z $\vec{x}'$ ($\vec{z}'$) označimo vektor, ki je v vseh komponentah enak $\vec{x}$ ($\vec{z}$), z izjemo i-te komponente, ki jo nastavimo na $\mathcal{X}_{i,j}$.

Enačba (3.13) zahteva dodatno pojasnilo, saj preskok iz tretje v četrto vrstico ni povsem očiten. V tretji vrstici smo vsoto preko vseh vrstnih redov namenoma pomaknili na sam začetek, da postane bolj jasno, da se za vsak vrstni red sprehodimo preko vseh parov $(\vec{x}, \vec{y}) \in \mathcal{X}$. Za vsak tak par potem sestavimo nov vektor, ki ima do mesta i-tega atributa v tem vrstnem redu vrednosti enake $\vec{x}$, od tam naprej pa enake vrednosti kot $\vec{y}$. Nato pogledamo razliko med napovedjo za ta vektor z vrednostjo i-tega atributa enako j in napovedjo za ta vektor. Vprašajmo se, s kolikšno verjetnostjo se pri danem vrstnem redu nek določen vektor $\vec{z} \in \mathcal{X}$ ob vsem tem rezanju in sestavljanju pojavi znotraj oklepaja. Ne glede na to, kje odrežemo (t.j., ne glede na vrstni red), je verjetnost pojavitve $\vec{z}$ enaka verjetnosti, da se njegov del, ki pripada $\vec{x}$ pojavi pri $\vec{x}$, pomnoženo z verjetnostjo, da se njegov preostanek pojavi pri $\vec{y}$. Če predpostavimo, da je rez na $l-$tem mestu, potem je verjetnost pojavitve $\vec{z}$ enaka $p(x_1 = z_1, x_2 = z_2, ..., x_l = z_l)p(y_{l+1} = z_{l+1}, ..., y_n = z_n)$. Ker $\vec{x}$ in $\vec{y}$ izberemo neodvisno in s ponovno uporabo predpostavke, da so atributi popolnoma neodvisni, dobimo, da je verjetnost pojavitve $\vec{z}$ kar $p(\vec{z})$.

Podobno kot splošen prispevek vrednosti nekega atributa (glej enačbo (3.13)) lahko iz prispevka atributa k napovedi izpeljemo tudi pomembnost atributa v celoti. Najprej si poglejmo slučajno spremenljivko

$$\Lambda_i = f(\vec{x}_{x_i=\mathcal{X}_{i,j}}) - f(\vec{x}),$$

katere porazdelitev je odvisna od modela f in porazdelitve primerov. Matematično upanje te slučajne spremenljivke

$$\begin{aligned} E[\Lambda_i] &= \sum_{j=1}^{|\mathcal{X}_i|} p'(\mathcal{X}_i = \mathcal{X}_{i,j})\psi_{i,j} \\ &= \sum_{j=1}^{|\mathcal{X}_i|} p'(\mathcal{X}_i = \mathcal{X}_{i,j}) \sum_{\vec{x}\in\mathcal{X}} p'(\vec{x}) \left(f(\vec{x}_{x_i=\mathcal{X}_{i,j}}) - f(\vec{x})\right) \end{aligned} \tag{3.14}$$

lahko preprosteje opišemo kot postopek, kjer najprej naključno izberemo neko vrednost i-tega atributa in nek primer $\vec{x}$. Nato pogledamo razliko, ki jo povzročimo, če vrednost i-tega atributa v $\vec{x}$ nastavimo na izbrano vrednost. Ker nas pri pomembnosti atributa predznak več ne zanima in ne želimo, da se nam pomemben atribut povpreči v 0, bomo podobno kot pri FIRM pomembnost atributa definirali kot $Var[\Lambda_i]$.

Podobno kot v podpoglavju 3.3 lahko tudi za aproksimacijo enačb (3.13) in (3.14) razvijemo ustrezna algoritma. Algoritem 4 je namenjen splošnemu prispevku vrednosti atributa, Algoritem 5 pa pomembnosti celotnega atributa.

Algoritem 4 Algoritem za aproksimacijo splošnega prispevka j-te vrednosti i-tega atributa Λ_i za model f. Vzorčimo m vzorcev.

$\psi_{i,j} \leftarrow 0$
for 1 to m **do**
  naključno izberemo $\vec{x} \in \mathcal{X}$
  sestavimo nov primer
  $\vec{b} \leftarrow$ vrednost i-tega atributa nastavimo na $\mathcal{X}_{i,j}$, preostale vrednosti kot v $\vec{x}$
  $\psi_{i,j} \leftarrow \psi_{i,j} + f(\vec{b}) - f(\vec{x})$
end for
$\psi_{i,j} \leftarrow \frac{\psi_{i,j}}{m}$

Algoritem 5 Algoritem za aproksimacijo pomembnosti i-tega atributa za model f. Vzorčimo m vzorcev.

$Var[\Lambda_i] \leftarrow 0$
for 1 to m **do**
  naključno izberemo vrednost i-tega atributa $\mathcal{X}_{i,j} \in \mathcal{X}_i$
  naključno izberemo $\vec{x} \in \mathcal{X}$
  sestavimo nov primer
  $\vec{b} \leftarrow$ vrednost i-tega atributa nastavimo na $\mathcal{X}_{i,j}$, preostale vrednosti kot v $\vec{x}$
  $Var[\Lambda_i] \leftarrow Var[\Lambda_i] + (f(\vec{b}) - f(\vec{x}))^2$
end for
$Var[\Lambda_i] \leftarrow \frac{Var[\Lambda_i]}{m}$

3.5 Pomembnejše lastnosti metode

V tem podpoglavju predstavimo še nekaj drugih lastnosti metode, ki nam pomagajo pri boljšem razumevanju metode in sorodnih metod ter dodatno utemeljijo praktično rabo metode.

3.5.1 Nov pogled na pristope z marginalnim prispevkom

Pokazali smo, da predlagana metoda odpravlja pomanjkljivosti obstoječih univerzalnih metod, ki smo jih izpostavili v 2. poglavju. Opisani pojmi iz teorije iger nam omogočajo, da bolj jasno opredelimo vzrok.

Najprej se spomnimo marginalnega prispevka, ki predstavlja najpogostejši pristop k računanju prospevka nekega atributa k napovedi modela v določenem primeru:

$$\text{marginal}\varphi_i = f(\vec{x}) - f(\vec{x} \setminus i).$$

Poglejmo si tako definirane prispevke z vidika Shapleyevih štirih aksiomov. Najprej poglejmo aksiome 2 do 4, nato pa še aksiom učinkovitosti.

V kontekstu marginalnega prispevka drugi aksiom pravi sledeče. Če za dva atributa i in j velja $f(\vec{x} \setminus \{i\}) = f(\vec{x} \setminus \{j\})$, potem imata atributa i in j enak marginalni prispevek[2]. Ta trditev drži.

Tretji aksiom pravi, da $f(\vec{x}) = f(\vec{x} \setminus x_i) \Rightarrow \text{marginal}\varphi_i = 0$. Tudi ta trditev očitno drži.

Četrti aksiom zahteva par iger f_1, f_2. Pokazati moramo, da velja $\text{marginal}\varphi_i(f_1+f_2) = \text{marginal}\varphi_i(f_1) + \text{marginal}\varphi_i(f_2)$, kjer je $(f_1+f_2)(S) = f_1(S) + f_2(S)$. $\text{marginal}\varphi_i(f_1+f_2)$ je enako $f_1(\vec{x}) - f_1(\vec{x} \setminus i) + f_2(\vec{x}) - f(\vec{x} \setminus i) = \text{marginal}\varphi_i(f_1) + \text{marginal}\varphi_i(f_2)$. Tudi ta aksiom drži.

Preostane prvi aksiom, aksiom učinkovitosti. Aksiom pravi, da mora biti vsota prispevkov enaka vrednosti velike koalicije. V kontekstu marginalnega prispevka je prispevek enega atributa razlika med napovedjo in napovedjo brez atributa, prispevek vseh atributov pa razlika med napovedjo in napovedjo brez vseh atributov.

Aksiom pravi, naj bo $\sum_{i \in N} \text{marginal}\varphi_i(f) = f(\vec{x}) - f(\varnothing)$. V 2. poglavju smo pokazali, da to drži v primeru, ko je f aditiven model. Marginalni prispevek v primeru aditivnih modelov izpolnjuje vse štiri aksiome. Kaj pa če f ni aditiven? Spomnimo se primera disjunkcije dveh binarnih atributov, kjer je $\sum_{i \in N} \text{marginal}\varphi_i(f) = 0$, čeprav vemo, da je $f(\vec{x}) - f(\varnothing) > 0$. V splošnem marginalni prispevek ne izpolnjuje aksioma učinkovitosti.

Kaj to pomeni, gledano s praktičnega vidika? Ker marginalni prispevek izpolnjuje aksiome 2-4, vemo, da bodo nepomembni atributi dobili prispevek 0 in atributi, ki prispevajo enako, enak prispevek. Ni bojazni, da bi nepomemben atribut označili za pomembnega.

Težave nastanejo pri relativizaciji rezultatov in primerjavi z drugimi primeri. Primer z disjunkcijo je tak, da atributa dobita manj, kot bi si zaslužila. Pri primeru s konjunkcijo

[2]Nekoliko daljše, a bolj v duhu aksioma, bi napisali $f((\vec{x} \setminus \{i,j\}) \cup j) = f((\vec{x} \setminus \{i,j\}) \cup i)$

dveh atributov pa je težava ravno nasprotna. $f(1 \wedge 1) = 1, f(?\wedge?) = 0.25$,in $f(1\wedge?) = f(? \wedge 1) = 0.5$. Marginalni prispevek vsakega atributa je 0.5, skupna razlika med napovedjo in povprečno napovedjo pa 0.75. Atributa dobita 0.25 več, kot je vrednost njune koalicije. Gledano iz ekonomskega stališča, je marginalni prispevek problematičen, saj je v večini primerov nesprejemljivo (nemogoče), da bi si razdelili manj (več) denarja, kot znaša skupna vsota denarja. Na enak način lahko tudi za prispevke metode FIRMΛ preverimo, da izpolnjuje vse aksiome razen prvega.

3.5.2 Ekvivalenca pri razlagi aditivnih modelov

V 2. poglavju smo obravnavali več specifičnih aditivnih modelov in splošnih metod za razlago aditivnih modelov. Ugotovili smo, da lahko aditivne modele razlagamo na učinkovit in razumljiv način. Vprašanje, zakaj bi za aditivne modele sploh uporabljali predlagano razlago in ne raje kar obstoječih metod, je povsem na mestu. Pri odgovoru nam bo pomagal naslednji izrek.

Izrek 3.4. *Naj funkcije $\psi_i(x), i = 1..n$ predstavljajo splošen prispevek vrednosti atributov za model f v kontekstu p. Če je model f aditiven, potem za vsak atribut i in njegovo vrednost $x \in \Re$ velja $\psi_i(x) = f_i(x) - E[f_i]$.*

Dokaz.

$$\psi_i(x) = \sum_{\vec{z}\in\mathcal{X}} p(\vec{z})(f(\vec{z}_{z_i=x}) - f(\vec{z})) =$$

$$= \sum_{\vec{z}\in\mathcal{X}} p(\vec{z})(f_i(x) - f_i(z_i)) =$$

$$= \sum_{\vec{z}\in\mathcal{X}} p(\vec{z}) f_i(x) - \sum_{\vec{z}\in\mathcal{X}} p(\vec{z}) f_i(z_i) =$$

$$= f_i(x) - E[f_i].$$

□

Izrek 3.4 nam pove, da aditivne modele s predlagano metodo razlagamo na enak način kot obstoječi pristopi. Natančneje, predlagana metoda je dekompozicija aditivnega modela na funkcije posameznih atributov. Spomnimo se, da aditivni model zapišemo kot $f(\vec{x}) = \sum_{i=1}^{n} \psi_i(x_i) + E[f]$, kjer je $E[f]$ od konteksta odvisna konstanta.

3.5.3 Povezava med kvaliteto razlage in kvaliteto napovedi

Prispevki atributov nam povejo, kako atributi vplivajo na napoved modela. Ko razlagamo uspešen model, nam razlaga pove tudi nekaj o neznanemu konceptu, ki ga modeliramo. In obratno, razlaga slabega modela nam ne povede veliko o neznanih konceptih.

Optimalno razlago neznanaga koncepta opredelimo kot razlago optimalnega modela, ki za vsak primer $\vec{x}$ napove $f_{\mathrm{OPT}}(\vec{x}) = E[g(\vec{x})]$. Optimalni model je v praksi redko dosegljiv, a intuitivno bi pričakovali, da model, za katerega dobimo razlago podobno optimalni, tudi podobno točno napoveduje primere. V ta namen definiramo razdaljo med dvema razlagama kot vsoto absolutnih razlik med komponentami razlag φ_A in φ_B, pri čemer upoštevamo tudi izhodišči $E[f_A]$ in $E[f_B]$:

$$d(\varphi_A, \varphi_B) = \sum_i |\varphi_{A,i} - \varphi_{B,i}| + |E[f_A] - E[f_B]|.$$

V dveh prejšnjih študijah smo na umetno generiranih množicah podatkov empirično preverjali, če razdalja do optimalne razlage pada z naraščanjem kvalitete modela (in obratno) [95, 98]. Rezultati so podprli hipotezo, saj je bil povprečni korelacijski koeficient med razdaljo (tam evklidsko) in napako (MSE verjetnostnih napovedi) preko vseh množic večji od 0.9. Uporaba umetnih množic je bila nujna, saj na realnih množicah ne poznamo dejanskih konceptov in ne moremo natančno opredeliti optimalnega modela.

V tem doktorskem delu omenjenega eksperimenta ne bomo ponovili, temveč se problema lotilmo teoretično. Postrežemo z nekaj ugotovitvami, ki so v pomoč pri razumevanju zveze med kvaliteto razlage in kvaliteto modela.

Izrek 3.5. *Če sta $f_A(\vec{x})$ in $f_B(\vec{x})$ napovedi dveh modelov za primer $\vec{x} \in \mathcal{X}$ ter φ_A in φ_B vektorja prispevkov za pripadajoči razlagi v kontekstu p, potem velja*

$$d(\varphi_A, \varphi_B) \geq |f_A(\vec{x}) - f_B(\vec{x})|.$$

Dokaz. Upoštevajmo lastnost predlagane metode, da je vsota prispevkov enaka razliki med napovedjo in pričakovano napovedjo v danem kontekstu.

$$d(\vec{\varphi}_A), \vec{\varphi}_B) \geq |f_A(\vec{x}) - f_B(\vec{x})|$$

$$\sum_i |\varphi_{A,i} - \varphi_{B,i}| + |E[f_A] - E[f_B]| \geq |\sum_i \varphi_{A,i} + E[f_A] - \sum_i \varphi_{B,i} - E[f_B]|$$

$$\sum_i |\varphi_{A,i} - \varphi_{B,i}| + |E[f_A] - E[f_B]| \geq |(\sum_i \varphi_{A,i} - \sum_i \varphi_{B,i}) + (E[f_A] - E[f_B])|$$

Ker $|a| + |b| \geq |a + b|$, kvečjemu povečamo desno stran, če jo preoblikujemo:

$$\sum_i |\varphi_{A,i} - \varphi_{B,i}| + |E[f_A] - E[f_B]| \geq |\sum_i \varphi_{A,i} - \sum_i \varphi_{B,i}| + |E[f_A] - E[f_B]|$$

$$\sum_i |\varphi_{A,i} - \varphi_{B,i}| \geq |\sum_i (\varphi_{A,i} - \varphi_{B,i})|$$

Kar drži, saj je $|a - b| \geq a - b$. □

Na podlagi tega izreka lahko rečemo, da bližje kot sta razlagi, bližje bosta tudi napovedi dveh modelov v istem kontekstu. V ekstremnem primeru enakih razlag, bosta enaki tudi napovedi.

Kaj pa obratno? Hitro lahko najdemo primer, ko imata modela enako napoved, nimata pa enake razlage. Na primer, $\mathcal{X} = [-1, 1]$, $E[X_1] = 0$, $f_A(x_1) = 0$ in $f_A(x_1) = x_1$. Pri $x_1 = 0$ bosta imela oba modela enako napoved, razlagi pa bosta različni!

3.5.4 Povezava med razporeditvijo varianc in težavnostjo razlage

V tem poglavju smo pokazali sledeče. Če želimo pri aproksimaciji doseči, da bo za prispevke vseh atributov enako verjetno, da bo napaka manjša od e, moramo vzorčiti sorazmerno variancam σ_i^2. Če je to naš kriterij, potem težavnost razlage nekega modela na neki množici (t.j., število vzorcev, ki jih potrebujemo, da napako znižamo pod nek nivo), opišemo z $\overline{\sigma^2} = \sum_{i=1}^{n} \sigma_i^2$. Če imata dva različna para model/množica enak $\overline{\sigma^2}$, sta oba enako težavna za razlago. Z drugimi besedami, za dano pričakovano napako bomo pri obeh potrebovali enako mnogo vzorcev.

Po drugi strani pričakovano vsoto absolutnih napak preko vseh prispevkov pri danem skupnem številu vzorcev m minimiziramo z razporeditvijo vzorcev, ki minimizira vsoto $\tau(\vec{m}, \vec{\sigma}^2) = \sum_{i=1}^{n} \sqrt{\frac{\sigma_i^2}{m_i}}$. Opazimo, da pri tem kriteriju dva različna para model/množica z enako $\overline{\sigma^2}$ nista nujno enako težavna za razlago. Težavnost postane odvisna ne le od vsote varianc, temveč od tega, kako je vsota porazdeljena med atribute. Na tem mestu si lahko zastavimo vprašanje, ki ima tudi praktičen pomen. Katere porazdelitve varianc so bolj težavne za razlago − če imamo le nekaj atributov z visoko varianco ali večje število atributov s sorazmerno nižjo varianco?

Imamo n atributov in vzorčimo $m \geq n$ vzorcev. Brez škode za splošnost lahko privzamemo, da je vsota varianc enaka $\sigma_1^2 + \sigma_2^2 + ... + \sigma_n^2 = 1$ in $1 \geq \sigma_1^2 \geq \sigma_2^2 \geq ... \geq \sigma_n^2 > 0$ ter razporeditev vzorcev $m = m_1 + m_2 + ... + m_n$. Najprej pokažimo, da je najbolj ugodna razporeditev varianc takrat, ko imamo en sam atribut (n = 1).

Dokaz. Če imamo samo en atribut z neničelno varianco, potem vsoto $\tau(\langle m_1 \rangle, \langle 1 \rangle) = \frac{1}{\sqrt{m_1}}$ minimiziramo z $m_1 = m$. Pri tem za dani m dobimo minimum $\frac{1}{\sqrt{m}}$.
Sedaj ponovno poglejmo vsoto $\tau(\langle m_1, ..., m_n \rangle, \langle \sigma_1^2, ...\sigma_n^2 \rangle)$ pri nekem n. Ker je $m \geq m_i$ in $\sigma_i^2 \geq 0$, velja:

$$\frac{\sqrt{\sigma_1^2}}{\sqrt{m_1}} + \frac{\sqrt{\sigma_2^2}}{\sqrt{m_2}} + ... + \frac{\sqrt{\sigma_n^2}}{\sqrt{m_n}} \geq \frac{\sqrt{\sigma_1^2}}{\sqrt{m}} + \frac{\sqrt{\sigma_2^2}}{\sqrt{m}} + ... + \frac{\sqrt{\sigma_n^2}}{\sqrt{m}} = \frac{\sqrt{\sigma_1^2}+\sqrt{\sigma_2^2}+...+\sqrt{\sigma_n^2}}{\sqrt{m}}.$$

Če upoštevamo $\sqrt{\sigma_1^2} + \sqrt{\sigma_2^2} + ... + \sqrt{\sigma_n^2} \geq 1$, dobimo:

$$\frac{\sqrt{\sigma_1^2}}{\sqrt{m_1}} + \frac{\sqrt{\sigma_2^2}}{\sqrt{m_2}} + ... + \frac{\sqrt{\sigma_n^2}}{\sqrt{m_n}} \geq \frac{\sqrt{\sigma_1^2}}{\sqrt{m}} + \frac{\sqrt{\sigma_2^2}}{\sqrt{m}} + ... + \frac{\sqrt{\sigma_n^2}}{\sqrt{m}} = \frac{\sqrt{\sigma_1^2}+\sqrt{\sigma_2^2}+...+\sqrt{\sigma_n^2}}{\sqrt{m}} \geq \frac{1}{\sqrt{m}} \qquad \square$$

Z drugimi besedami, pri dani vsoti varianc in številu vzorcev bomo najmanjšo pričakovano skupno napako dosegli takrat, ko bomo imeli en sam pomemben atribut.

Sedaj rezultat posplošimo na primer, ko imamo n enako pomembnih atributov: $\sigma_i^2 = \frac{1}{n}$, $i = 1..n$. Vsoto, ki jo minimiziramo, lahko zapišemo kot

$$\frac{\sqrt{\sigma_1^2}}{\sqrt{m_1}} + \frac{\sqrt{\sigma_2^2}}{\sqrt{m_2}} + ... + \frac{\sqrt{\sigma_n^2}}{\sqrt{m_n}} = \frac{\sqrt{\frac{1}{n}}}{\sqrt{m_1}} + \frac{\sqrt{\frac{1}{n}}}{\sqrt{m_2}} + ... + \frac{\sqrt{\frac{1}{n}}}{\sqrt{m_n}},$$

pri čemer minimum dosežemo z enakomerno razporeditvijo vzorcev

$$\frac{\sqrt{\frac{1}{n}}}{\sqrt{\frac{m}{n}}} + \frac{\sqrt{\frac{1}{n}}}{\sqrt{\frac{m}{n}}} + ... + \frac{\sqrt{\frac{1}{n}}}{\sqrt{\frac{m}{n}}} = n\sqrt{\frac{1}{m}}.$$

Ko so variance enakomerno razporejene med atribute, je manj težavno razlagati pri manjšem številu atributov. Z drugimi besedami, za dosego enake pričakovane skupne absolutne napake bomo pri manj atributih potrebovali manj vzorcev, čeprav bo vsota varianc enaka.

Kje pa se po težavnosti nahajajo primeri, ko variance niso enakomerno razporejene? Pokažimo, da je pri danem številu atributov n in vzorcev $m \geq n$ najbolj težavna ravno enakomerna razporeditev varianc. Vzemimo neko poljubno razporeditev neničelnih varianc, katerih vsota je ena, in zapišimo vsoto, ki jo minimiziramo:

$$\frac{\sqrt{\sigma_1^2}}{\sqrt{m_1}} + \frac{\sqrt{\sigma_2^2}}{\sqrt{m_2}} + ... + \frac{\sqrt{\sigma_n^2}}{\sqrt{m_n}}$$

Ne glede na razporeditev varianc, lahko vzorce razporedimo enakomerno

$$\frac{\sqrt{\sigma_1^2}}{\sqrt{\frac{m}{n}}} + \frac{\sqrt{\sigma_2^2}}{\sqrt{\frac{m}{n}}} + ... + \frac{\sqrt{\sigma_n^2}}{\sqrt{\frac{m}{n}}} = \sqrt{n}\frac{1}{\sqrt{m}}$$

To pomeni, da ne glede na razporeditev varianc že z enakomerno razporeditvijo dosežemo boljši, kvečjemu enako dober, rezultat, kot bi ga lahko pri enakomerno razporejenih variancah.

Poglavje 4

Poskusi

V tem poglavju opišemo poskuse, ki smo jih izvedli, in njihove rezultate. Najprej predstavimo množice podatkov in učne algoritme, ki smo jih uporabili. Nato si izmenoma sledijo opisi poskusov in pripadajoči rezultati. Poskuse razdelimo na dva dela. V prvem delu se osredotočimo na merjenje časov izvajanja in varianco vzorcev. Naš namen je ugotoviti, če je metoda dovolj učinkovita za praktično rabo. V drugem delu s pomočjo vizualnega pregleda ocenimo uporabnost predlagane metode z vidika razumljivosti razlag.

4.1 Izbrane množice podatkov in učni algoritmi

Večino poskusov smo izvedli na klasifikacijskih in regresijskih množicah podatkov, ki jih najdemo v tabelah 4.1 in 4.2. Množice razdelimo na dve skupini. V prvo skupino spadajo umetne množice, ki so označene bodisi s predpono "c" (klasifikacija) bodisi s predpono "r" (regresija). Te množice smo ustvarili z namenom, da na kontroliran način preverimo uspešnost predlagane metode pri različnih konceptih in posebnih primerih. Umetne množice podatkov bolj podrobno opišemo kasneje. V drugo skupino spadajo splošno znane množice podatkov[1]. Te množice smo vključili z namenom, da ocenimo delovanje metode na primerih, s katerimi se srečujemo v vsakdanji praksi. Množice so na voljo na spletni strani Weke (`http://www.cs.waikato.ac.nz/ml/weka/`). Večino teh množic najdemo tudi v repozitoriju UCI [5].

Umetno generirane množice podatkov nam omogočajo, da preverimo delovanje predlagane metode na konceptih, kjer druge metode odpovejo (ekskluzivni-ali, disjunkcija). Prav tako lahko preverimo njeno delovanje na nekaterih drugih konceptih, ki jih pogosto srečamo v praksi (pogojno neodvisni atributi, redundantni atributi, naključni atributi, itd...). Nekatere izmed teh množic podatkov, predvsem klasifikacijske, so že bile uporabljene v prejšnjih študijah [82, 95, 98, 96], čeprav v nekoliko spremenjeni obliki. Če ni posebej omenjeno, so vsi numerični atributi omejeni na interval med 0 in 1, preostali atributi pa so binarni. Atributi, ki niso eksplicitno omenjeni v opisu, so nepomembni za

[1] Večina teh množic je sicer realnih, znane množice monks1-3 pa so umetne

Tabela 4.1: Osnovni podatki o uporabljenih klasifikacijskih učnih množicah: število primerov (#P), skupno število atributov (#A), število nominalnih atributov (#Nom), število numeričnih atributov (#Num).

Ime	#P	#A	#Nom.	#Num	Opis
cChess	2000	4	0	4	Barva polja na šahovnici.
cCondInd	2000	8	8	0	Pogojno neodvisni atributi.
cCross	2000	6	0	6	Soda ali liha kvadranta.
cDisjunctB	2000	5	3	2	Disjunkcija z binarnimi atributi.
cDisjunctN	2000	5	0	5	Disjunkcija z numeričnimi atributi.
cGroup	2000	4	0	4	Gruče.
cRandom	2000	4	0	4	Atributi so naključni in nepomembni.
cRedundant	2000	5	0	5	Disjunkcija z redundantnimi atributi.
cSphere	2000	5	0	5	Ali leži točka znotraj krogle?
cXor	2000	6	6	0	Ekskluzivni-ali.
anneal	898	38	32	6	Ohlajanje jekla.
breast-cancer(Lj)	286	9	9	0	Napovedovanje ponovitve raka.
hepatitis	155	19	13	6	Smrtnost zbolelih za hepatitisom.
iris	150	4	0	4	Vrsta rastline.
monks1	432	6	6	0	Znana umetna množica (glej [89]).
monks2	432	6	6	0	Znana umetna množica (glej [89]).
monks3	432	6	6	0	Znana umetna množica (glej [89]).
mushroom	8124	22	22	0	Užitnost gob.
nursery	12960	8	8	0	Ocenjevanje prijav v vrtec.
soybean	683	35	35	0	Diagnoza bolezni rastline soje.
zoo	101	16	15	1	Razred živali.

napovedovanje:

cChess. Podatki opisujejo šahovnico 4×4 na enotskem kvadratu $A_1 \times A_2$. Bela polja predstavljajo razred 1, črna pa razred 0.

cCondInd. Obe razreda (0 in 1) sta enako verjetna. Prvi štirje atributi A_{1-4} so povezani z razredom. Vrednost atributa A_1 je enaka vrednosti razreda v 60% primerov, A_2 v 70%, A_3 v 80% in A_4 v 90% vseh primerov.

cCross. Vrednost razreda je 1, ko je $(A_1) \cdot (A_2) > 0.5$, in 0, v vseh ostalih primerih.

cDisjunctB. Prvi trije atributi so binarni in vsaj eden izmed njih mora biti enak 1, da je razred enak 0.

cDisjunctN. Prvi trije numerični atributi so pomembni za napovedovanje razreda. Razred je enak 1, če je izpolnjen vsaj eden izmed pogojev: $A_1 > 0.5$, $A_2 > 0.7$ ali $A_3 < 0.4$.

cGroup. Atributa A_1 in A_2 predstavljata osi v kartezičnem prostoru. Enotski kvadrat razdelimo na devet enakih kvadratov, vsak kvadrat pa predstavlja enega izmed treh možnih razrednih vrednosti. Razredne razporedimo tako, da samo z poznavanjem ene izmed koordinat ni mogoče ugotoviti, kateremu razredu pripada primer.

cRandom. Vsi atributi so nepomembni za vrednost razreda.

cRedundant. Enako kot cDisjunctN, le da atributa A_4 in A_5 nista več nepomembna, temveč sta kopiji atributov A_1 in A_2.

cSphere. Prvi trije atributi opisujejo položaj primera v enotski kocki. Razredna vrednost nam pove, če točka leži zunaj ali znotraj krogle, ki je včrtana v enotsko kocko.

cXor. Razredna vrednost je enaka parnosti prvih treh atributov. Razredne vrednosti vsebujejo 10% šuma.

rDisjunctB. Enako kot cDisjunctB, le da binarni razred obravnavamo kot numerično spremenljivko.

rDisjunctN. Enako kot cDisjunctN, le da binarni razred obravnavamo kot numerično spremenljivko.

rLinear. Linearna funkcija $A_1 - 2 \cdot A_2 + 1.5 \cdot A_3$.

rLinNoisy. Enako kot rLinear, le da je atributom dodan šum. Vrednosti atributa prištejemo naključno izbrano vrednost z intervala $\in [-0.1, 0.1]$.

rLocLinear. Če je $A_1 > 0.5$, potem $2 \cdot A_2 - A_3$. Drugače $-0.5 \cdot A_4 - 2 \cdot A_5$.

rNonLinPoly. Nelinearna funkcija $8 \cdot (A_1 - 0.6)^3 + 5 \cdot (A_1 - 0.6)^2 + 0.3 \cdot A_2$.

rNonLinTrig. Nelinearna funkcija $sin(2\pi \cdot A_1) + A_2 - A_3$.

rRandom. Enako kot cRandom, le da binarni razred obravnavamo kot numerično spremenljivko.

rRedundant. Enako kot rDisjunctN, le da atributa A_4 in A_5 nista več nepomembna, temveč sta kopiji atributov A_1 in A_2.

rXor. Enako kot cXor, le da binarni razred obravnavamo kot numerično spremenljivko.

4.1.1 Posebne množice podatkov

Poleg opisanega nabora množic podatkov, smo uporabili še dve manjši skupini množic podatkov. Prva skupina je sestavljena iz treh množic testA, testB in testC, ki opisujejo primer študenta na izpitu (glej uvodno poglavje). V vseh treh različicah je koncept, ki določa študentove možnosti, enak, množice pa imajo različno porazdeljene vrednosti atributov. Pri testA profesor vpraša le šestino študentov, pri testB devet od desetih študentov, pri testC pa so vse kombinacije atributov enako verjetne. Rezultate na teh množicah uporabimo za proučevanje vpliva spremembe konteksta na razlago.

Druga skupina sta množici datagen40 in linear50, ki smo ju uporabili za analizo povečevanja porabe časa pri povečevanju števila atributov. Množica *linear50* vsebuje 1000 primerov in ima 50 numeričnih atributov, ki so med seboj neodvisni in standardno normalno porazeljeneni. Gre za regresijski problem, katerega rešitev je linearna kombinacija atributov. Vsak drugi atribut ima koeficient enak 1, preostalim atributom pa smo koeficiente izbrali naključno iz podmnožice celih števil med -5 in 5.

Množica *datgen40* vsebuje 1000 primerov. Vsak primer ima 40 nominalnih atributov, vsak atribut 10 vrednosti (od a do j). Trideset atributov je nepomembnih, preostalih 10 atributov določa, kateremu izmed desetih razredov pripada primer:

Tabela 4.2: Osnovni podatki o uporabljenih regresijskih učnih množicah: število primerov (#P), skupno število atributov (#A), število nominalnih atributov (#Nom), število numeričnih atributov (#Num).

Ime	#P	#A	#Nom	#Num	Opis
rDisjunctB	2000	5	3	2	Disjunkcija z binarnimi atributi.
rDisjunctN	2000	5	0	5	Disjunkcija z numeričnimi atributi.
rLinear	2000	5	0	5	Linearni problem.
rLinNoisy	2000	5	0	5	Linearni problem s šumom.
rLocLinear	2000	5	0	5	Lokalno-linearen problem.
rNonLinPoly	2000	5	0	5	Polinom tretje stopnje.
rNonLinTrig	2000	5	0	5	Trigonometrična funkcija.
rRandom	2000	4	0	4	Atributi so naključni in nepomembni.
rRedundant	2000	5	0	5	Disjunkcija z redundantnimi atributi.
rXor	2000	6	6	0	Ekskluzivni ali.
autoMpg	398	7	3	4	Poraba goriva pri avtu.
bodyfat	252	14	0	14	Napovedovanje indeksa telesne mase.
concrete	1030	8	0	8	Kompresivna moč cementa.
elevators	16599	18	0	18	Upravljanje z letalom.
fishcatch	158	7	2	5	Napovedovanje teže ujete ribe.
fruitfly	125	4	2	2	Življenjska doba sadnih muh.
housing	506	13	1	12	Vrednost zemljišč v Bostonu.
machinecpu	209	6	0	6	Preformanse procesnih enot.
pollution	60	15	0	15	Onesnaženost zraka in smrtnost.
stock	950	9	0	9	Gibanje cene delnic.
wine	4898	11	0	11	Kvaliteta vina.
wisconsin	194	32	0	32	Napovedovanje ponovitve raka.

$$(A_{15} = j \wedge A_{31} = f \wedge A_{33} = f) \vee (A_{33} = a) \Rightarrow C_1$$
$$(A_{12} = j \wedge A_{21} = f \wedge A_{33} = f) \vee (A_{33} = i) \Rightarrow C_2$$
$$(A_{12} = d \wedge A_{21} = f \wedge A_{33} = j) \vee (A_{01} = c \wedge A_{23} = h \wedge A_{01} = g) \vee (A_{33} = c \wedge A_{01} = g) \Rightarrow C_3$$
$$(A_{12} = j \wedge A_{23} = i \wedge A_{33} = b) \Rightarrow C_4$$
$$(A_{12} = j \wedge A_{01} = b \wedge A_{33} = d) \Rightarrow C_5$$
$$(A_{03} = g \wedge A_{21} = j \wedge A_{33} = j) \Rightarrow C_6$$
$$(A_{04} = h \wedge A_{23} = a \wedge A_{33} = b) \vee (A_{33} = f \wedge A_{31} = b) \Rightarrow C_7$$
$$(A_{04} = e \wedge A_{23} = a \wedge A_{33} = e) \vee (A_{04} = h \wedge A_{31} = i \wedge A_{33} = e) \vee (A_{12} = h \wedge A_{33} = i) \Rightarrow C_8$$
$$(A_{15} = c \wedge A_{23} = f \wedge A_{33} = g) \Rightarrow C_9$$
$$(A_{12} = c \wedge A_{33} = i) \vee (A_{23} = a \wedge A_{33} = g) \vee (A_{21} = i \wedge A_{23} = h \wedge A_{33} = j) \Rightarrow C_{10}$$

Gre za klasifikacijski problem. Množico smo ustvarili z Mellijevim generatorjem umetih množic podatkov [66], ki omogoča generiranje atributov in logičnih pravil, ki jih povezujejo z razredom, pri čemer lahko nastavimo število in tip atributov, število primerov, kolikšen delež primerov pokrije vsako pravilo ter druge parametre.

4.1.2 Učni algoritmi

V poskuse smo vključili deset različnih klasifikacijskih in sedem različnih regresijskih modelov. Število različnih učnih algoritmov je manjše, saj so nekateri algoritmi uporabni tako v regresiji kot v klasifikaciji. Uporabili smo implementacije iz knjižnice orodja za podatkovno rudarjanje Weka [35]. Kjer ni posebej omenjeno, smo uporabili privzete nastavitve. Seznam uporabljenih učnih algoritmov najdemo v tabeli 4.3.

4.2 Časovna učinkovitost predlagane metode

S praktičnega vidika nas zanima predvsem, koliko časa bomo potrebovali za računanje razlage v določenem primeru oziroma kje so omejitve predlagane metode. Preden predstavimo rezultate za celoten nabor množic podatkov in modelov, si na dveh ilustrativnih množicah podatkov poglejmo najpomembnejše komponente časa računanja pri razlagi.

4.2.1 Ilustracija z naraščajočim številom atributov

Čas računanja razlage napovedi danega modela za primer iz neke množice podatkov je sestavljen iz dveh komponent. Prva je število vzorcev, ki jih potrebujemo, druga čas, ki ga potrebujemo za računanje enega vzorca. Potrebno število vzorcev ocenimo iz varianc $\sigma_i, \forall i$, čas pa je neposredno povezan s časom, ki ga model potrebuje za eno napoved.

Če smo bolj natančni, je čas računanja enega vzorca sestavljen iz dveh napovedi in ustreznega preoblikovanja primerov (glej algoritem 1). Na sliki 4.1(a) vidimo, kako se

Tabela 4.3: Seznam uporabljenih učnih algoritmov.

Algoritem	Opis
AdaBoostM1	Metaučenje z uporabo AdaBoost algoritma. Kot šibek model smo uporabili bodisi naivnega Bayesa bodisi odločitveno drevo.
Bagging	Metaučenje bagging Kot šibek algoritem smo uporabili bodisi odločitveno bodisi regresijsko drevo.
IBk	Metoda najbližjih sosedov je uporabna tako v klasifikaciji kot tudi v regresiji. Uporabili smo dve različici (k = 1 in k = 11)
J48	Algoritem za gradnjo odločitvenih dreves.
LinearRegression	Linearna regresija.
Logistic	Logistična regresija.
M5P	Algoritem za gradnjo regresijskih dreves.
MultilayerPerceptron	Umetne nevronske mreže smo uporabili tako v klasifikaciji kot tudi v regresiji. Uporabili smo en sam nivo skritih nevronov.
NaiveBayes	Osnovna različica algoritma naivnega Bayesa je uporabna za reševanje klasifikacijskih problemov.
RandomForest	Breimanovi naključni gozdovi [13]
SMO	Metoda podpornih vektorjev (SVM) za reševanje klasifikacijskih problemov. Izbrali smo polinom druge stopnje.
SVMreg	

čas računanja povečuje, ko modele gradimo na vedno večjem deležu atributov množice *datgen40*. Z izjemo umetne nevronske mreže (MultilayerPerceptron) se pri vseh algoritmih čas računanja modela povečuje kvečjemu linearno s številom atributov[2]. Torej $c \cdot O(n)$, pri čemer se konstanta c bistveno razlikuje od modela do modela.

Število vzorcev m_i, ki jih potrebujemo, da za i-ti atribut dosežemo želeno omejitev napake, je sorazmerno σ_i^2. Skupno število vzorcev $m \sum_{i=1}^{n} m_i$ je sorazmerno vsoti varianc oziroma $n \cdot \frac{1}{n} \sum_{i=1}^{n} \sigma_i^2 = n\overline{\sigma^2}$. Teoretične meje povprečne variance $\overline{\sigma^2}$ že poznamo. Če normaliziramo napovedi, je zgornja meja ena polovica, spodnja meja pa je v vsakem primeru 0. Pri klasifikaciji normalizacija ni potrebna, saj so vrednosti prispevkov že implicitno omejene med -1 in 1. Pri regresiji normaliziramo tako, da prispevke za dano množico in model delimo z absolutno največnjim prispevkom za to množico in model. Naj poudarimo, da to storimo zgolj za potrebe lažje primerjave varianc preko različnih modelov in množic podatkov. Tudi iz praktičnega vidika je tak način normalizacije sprejemljiv, saj gledamo varianco z vidika skale prispevkov pri vizualizaciji. Skala vizualizacije mora biti takšna, da lahko prikažemo prispevke med 0 in maksimalnim prispevkom, prispevki

[2]Vsaj za potrebe razpona števila atributov.

(razlike med prispevki), ki so tako majhni, da jih pri taki skali ne opazimo, nas običajno tudi ne zanimajo.

V praksi je $\overline{\sigma^2}$ odvisna od modela oziroma tega, česar se je model naučil. Slednje je običajno omejeno s kompleksnostjo koncepta, ki ga poskušamo modelirati. Vendar v splošnem ne moremo zagotoviti, da se model ne bo naučil bolj zapletenega koncepta oziroma se preveč prilagodil učni množici. Če pogledamo kvaliteto modelov (slika 4.2(b)) in povprečno varianco (slika 4.1(b)), vidimo, da se najbolje obnesejo trije modeli (odločitvena drevesa, boosting z odločitvenimi drevesi in logistična regresija), ki imajo na koncu tudi nizko povprečno varianco.

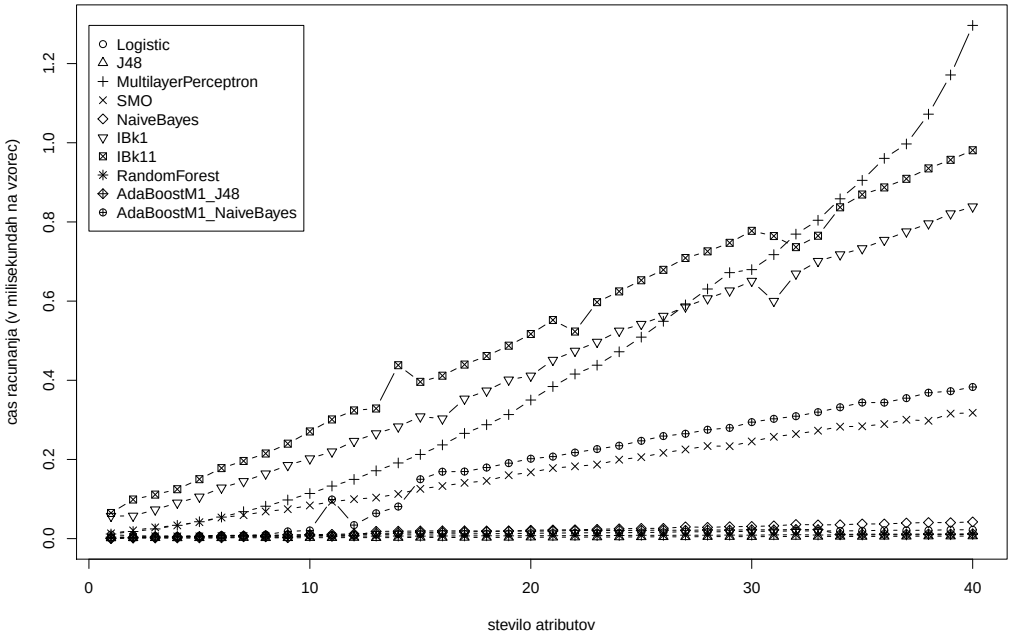
Ko pogledamo še minimalno varianco preko vseh atributov (glej sliko 4.2(a)) vidimo, da so le trije algoritmi zgradili model, ki vsaj nekaterih atributov ne upošteva (njihova varianca je 0). Neupoštevanje nekaterih atributov je pravilno, saj je le 10 od 40 atributov pomembnih za napovedovanje razreda.

Modeli, ki imajo višjo povprečno varianco od najboljšega modela a slabšo točnost, so se preveč prilagodili učni množici. Sem lahko uvrstimo modele AdaBoostM1NaiveBayes, MultilayerPerceptron (do 33 atributov) in IBk1. Slednji še posebej izstopa, saj je po kvaliteti najslabši model, a ima najvišjo minimalno varianco, ki je skoraj enaka povprečni. To pomeni, da so vsi atributi približno enako pomembni za napovedi tega modela. Ker gre za metodo k-najbližjih sosedov (k = 1), je tak rezultat pričakovan, saj je nagnjena k prevelikem prilagajanju. Še bolj preprost in jasen primer prevelikega prilagajanja učni množici *cRandom* je opisan v podpoglavju 4.3.

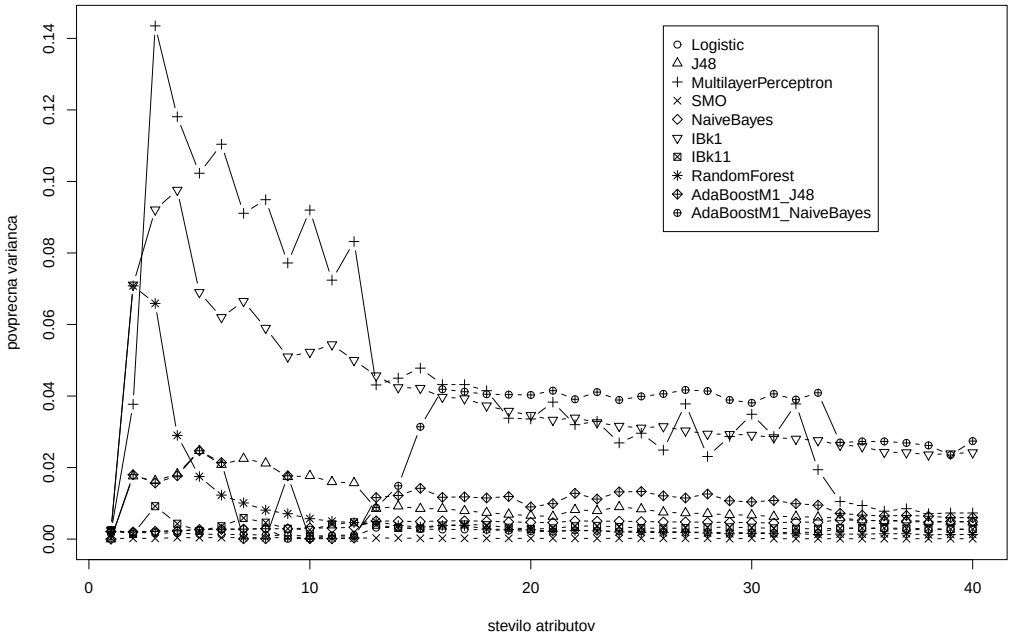
Časi računanja vzorca so pri linear50 podobni kot pri datgen40, le da obe metodi najbližjih sosedov še bolj izstopata. Napaka modelov pričakovano pada s številom atributov (glej sliko 4.3(a)), čeprav niso vsi modeli enako uspešni. Linearna regresija, regresijsko drevo in regresijski SVM pri 40 atributih dosežejo najnižjo napako, ki je sicer večja od nič, saj je na voljo le 40 od 50 pomembnih atributov. Glede na povprečno varianco preostale modele razdelimo na dve skupini: tiste, ki so se naučili premalo (Bagging, MultilayerPerceptron), in tiste, ki so se naučili preveč (IBK1, IBK11).

Ker smo izmerili čase računanja in povprečne variance, lahko ocenimo čas, ki ga potrebujemo za izračun razlage za en primer: $n \cdot \frac{1.96^2\overline{\sigma^2}}{(0.05)^2} \cdot$ število sekund / vzorec, kjer je n število atributov in $P(|e \leq 0.05|) = 0.95$ želena omejitev napake ($Z_{1-0.05} \approx 1.96$). Pri tem smo predpostavili, da ima vsak atribut varianco enako povprečni varianci preko vseh atributov. Iz grafov je razvidno, da to ne drži, a spomnimo se, da pri zagotavljanju enake omejitve napake razporeditev variance ni pomembna za določanje števila vzorcev, ki jih potrebujemo - pomembna je le vsota.

Na sliki 4.4 so prikazani izmerjeni časi računanja ene razlage za obe množici. Vidimo, da največ časa potrebujemo za razlago napovedi metod najbližjih sosedov. Kljub temu, da pri večjem številu atributov domene datgen40 največ časa potrebujemo za računanje napovedi nevronske mreže, je čas računanja razlage pri tem modelu nižji, ker ima nižjo povprečno varianco. Pri interpretaciji rezultatov v naslednjem podpoglavju bodimo pozorni tako na povprečno varianco kot tudi na čas računanja vzorca, čeprav slednji igra

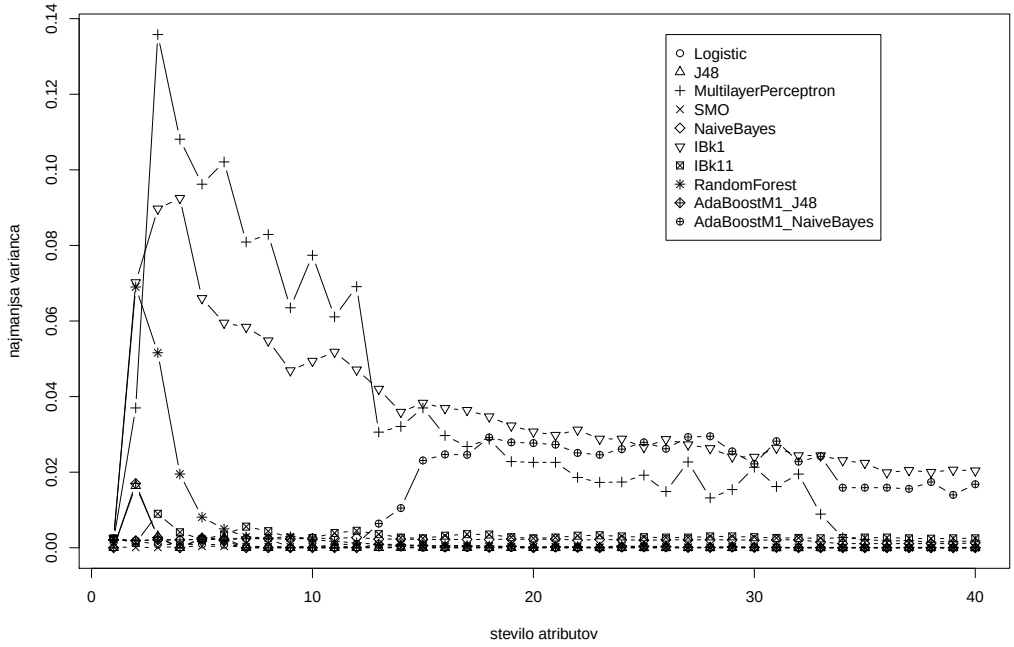


(a) časi računanja

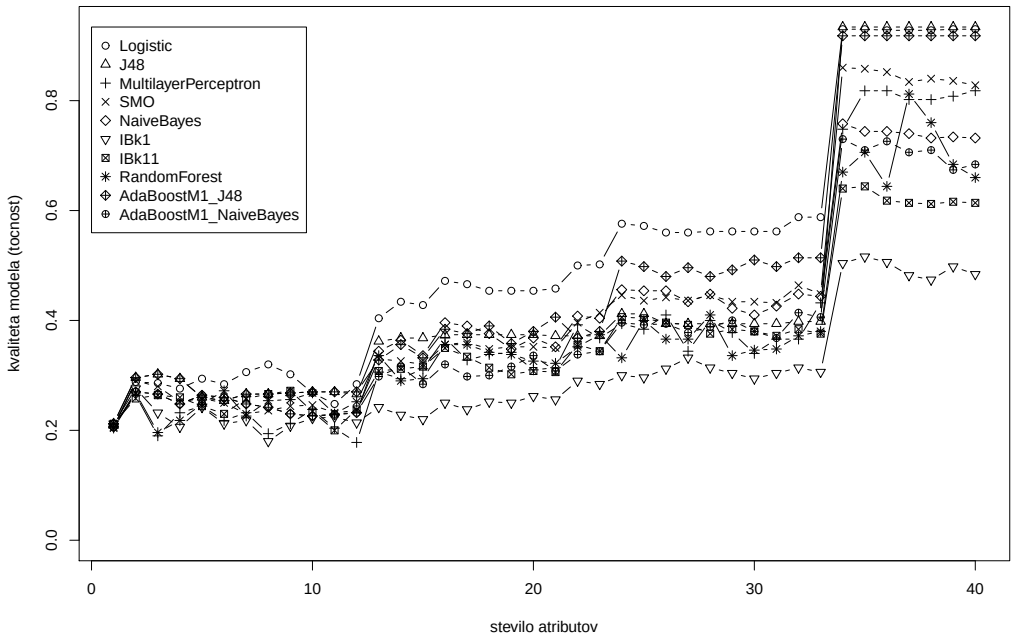


(b) povprečne variance

Slika 4.1: Časi računanja in povprečne variance za datgen40 in več učnih algoritmov. Vsaka točka predstavlja model zgrajen na podmnožici atributov določene velikosti.

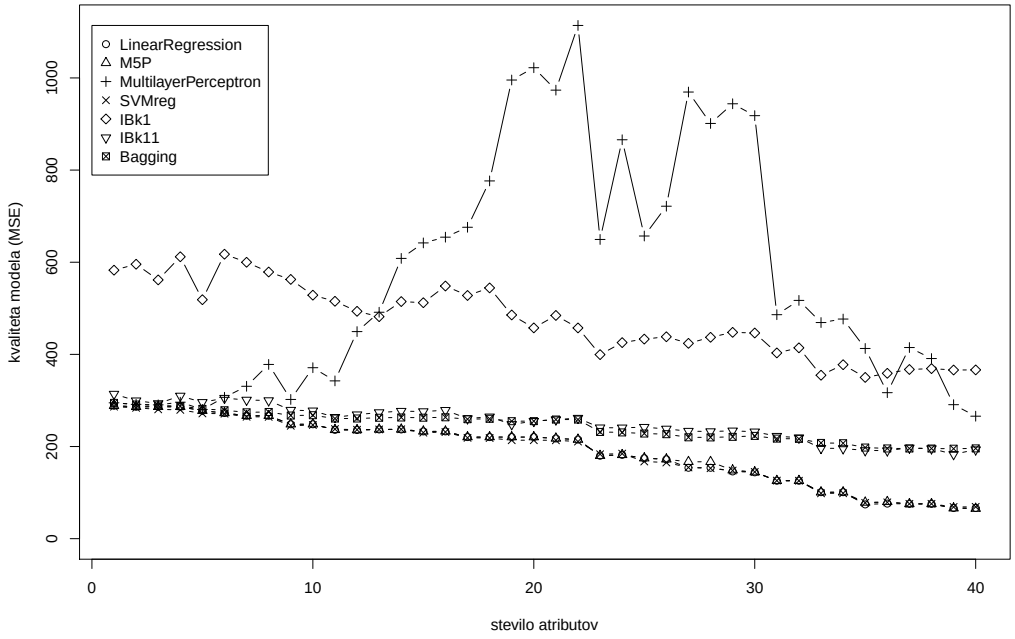


(a) najmanjše variance

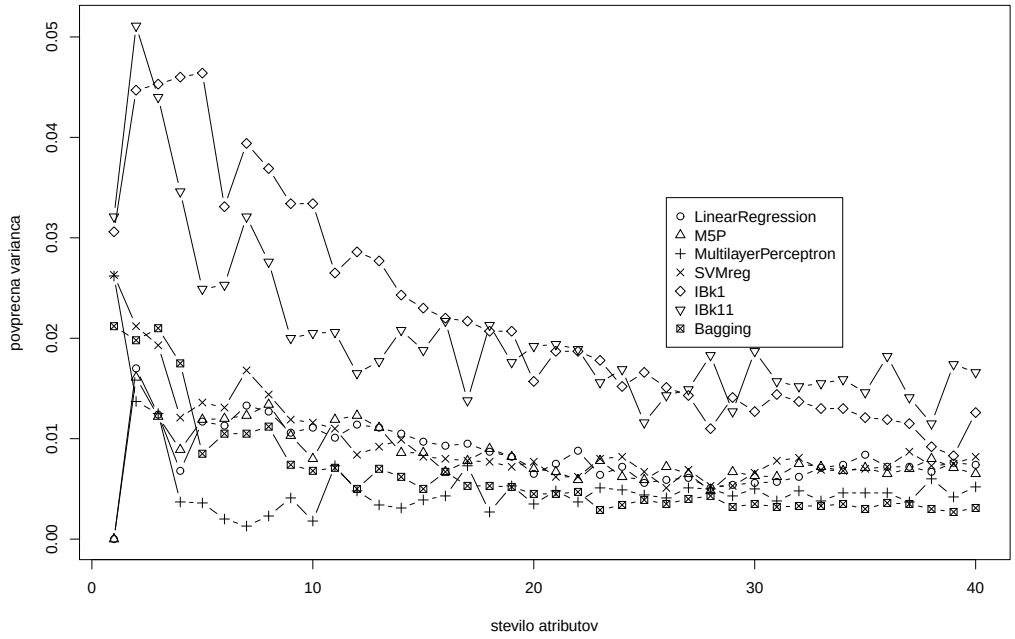


(b) točnosti modelov

Slika 4.2: Najmanjše izmerjene variance in kvaliteta modelov na množici datgen40. Vsaka točka predstavlja model, zgrajen na podmnožici atributov določene velikosti. Kvaliteto modelov smo ocenili na neodvisni testni množici.



(a) kvaliteta modelov



(b) najvecje variance

Slika 4.3: Kvaliteta modelov in povprečne variance za linear50 in več učnih algoritmov. Vsaka točka predstavlja model, zgrajen na podmnožici atributov določene velikosti. Zadnjih 10 atributov smo izpustili.

bolj pomembno vlogo.

4.2.2 Variance in časi računanja

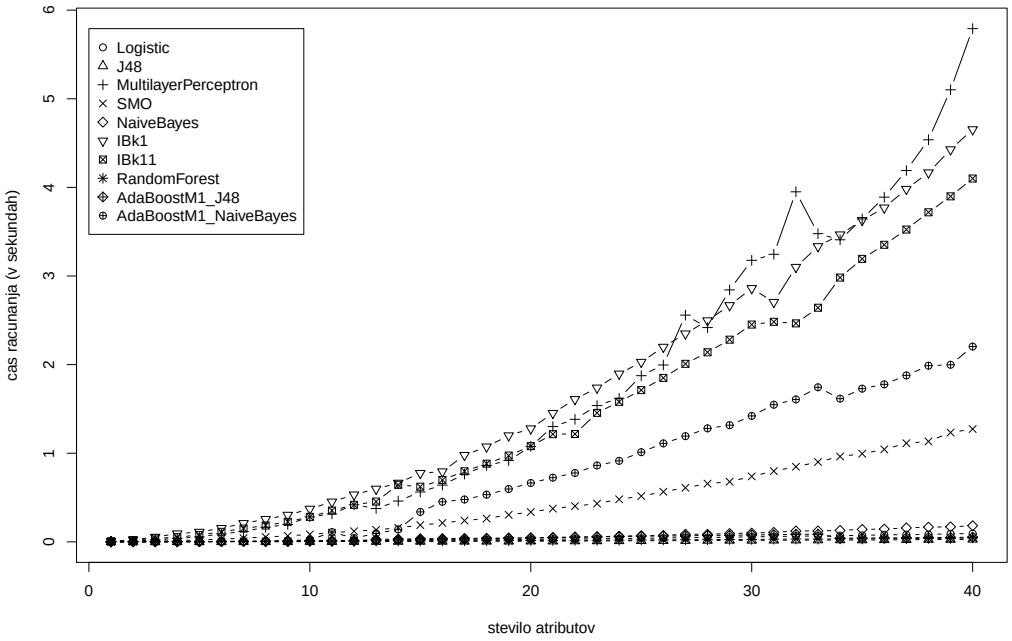
Na podlagi rezultatov iz prejšnjega podpoglavja lahko zaključimo, da lahko za datgen40 in linear50 v nekaj sekundah generiramo razlago za poljuben model iz našega nabora modelov. Vsaj za ti dve množici zaključimo, da lahko v praktičnem času izračunamo razlago tudi za več deset atributov. Poglejmo si, če tudi v praksi srečamo primerljive čase računanja in variance. Kot smo podrobneje opisali v prejšnjem poglavju, smo slednje pri regresijskih množicah računali iz normaliziranih prispevkov. S tem smo omogočili lažjo primerjavo variance preko različnih regresijskih množic in modelov. Povprečne variance, čase računanja in točnosti modelov za vse pare model/množica, najdemo v dodatku A. V tem podpoglavju povzamemo rezultate.

Vse rezultate, ki govorijo o kvaliteti posameznih modelov, smo pridobili z 10-kratnim prečnim preverjanjem. Kvaliteti modelov nismo posvečali posebne pozornosti, saj nam služi predvsem kot pomoč pri interpretaciji razlag. Edini cilj je bil dovolj velika pestrost, da lahko vizualno preverimo in primerjamo tako razlage dobrih modelov kot tudi razlage slabih modelov. Ta cilj smo izpolnili, saj pri vseh množicah najdemo uspešno in neuspešno naučene modele. V povprečju so najuspešnejši učni algoritmi, ki gradijo drevesa (naključni gozdovi za klasifikacijo in M5P za regresijo ter Bagging pri obeh), predvsem taki, ki napovedi večjega števila dreves združijo z metaučenjem bagging.

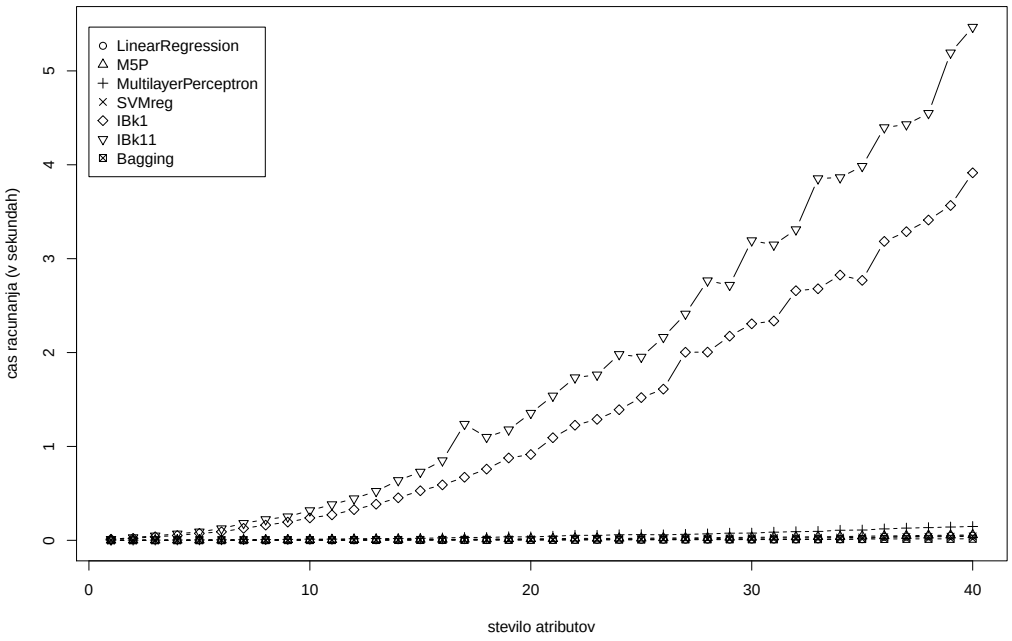
Časi računanja so podobni kot pri množicah datgen40 in linear50 pri ustreznem številu atributov. Čas računanja napovedi ne ovira praktične rabe na množicah z nekaj deset atributi. Izstopa metoda k-najbližjih sosedov. Izmed vseh je slednja edina, pri kateri je čas računanja močno odvisen tudi od števila učnih primerov. Časi računanja na množicah elevators, mushroom, nursery in wine so do šestkrat večji od časov računanja za datgen40 pri 40 atributih, kljub manjšemu številu atributov. Pri tem je vredno omeniti, da smo uporabili izčrpno iskanje najbližjih sosedov. Čase računanja bi zmanjšali z uporabo kd-dreves.

Tudi povprečne variance so podobne kot pri datgen40 in linear50. Kljub temu, da je čas računanja za en vzorec pri metodi najbližjih sosedov na množici mushroom (povprečna varianca 0.0172) skoraj 9 milisekund, je zaradi nizke povprečne variance skupni čas za računanje prispevkov približno 5 sekund (uporabili smo enako omejitev napake, kot pri ilustrativnem primeru iz prejšnjega podpoglavja). Če bi pred tem ocenili še variance s 50 vzorci za vsak atribut, bi temu času prišteli še slabih 10 sekund.

Nasploh so povprečne variance višje pri umetnih množicah podatkov, saj le-te v povprečju vsebujejo manj nepomembnih atributov in bolj izrazite koncepte. Najvišjo povprečno varianco (0.4056) smo izmerili na množici cRandom za metodo najbližjih sosedov, kjer je prišlo do prevelikega prilagajanja podatkom. Če odmislimo primere, kjer je prišlo do prevelikega prilagajanja, pa je najvišja povprečna varianca na monks2 za umetno nevronsko mrežo (0.2431). Gledano z vidika časa računanja razlage (povprečne variance



(a) datgen40



(b) linear50

Slika 4.4: Izmerjeni časi računanja razlage za eno napoved v odvisnosti od števila atributov za množici datgen40 in linear50.

pomnožimo s številom atributov), je monks2 1.5 krat bolj težaven za razlago od množice mushroom.

Za naš nabor množic in učnih algoritmov velja, da za računanje prispevkov potrebujemo kvejčemu nekaj sekund. Smiselnost računanja prispevkov atributov k napovedi za množice z več sto ali več tisoč atributi je vprašljiva že z vidika razumljivosti tolikšnega števila prispevkov. Množice podatkov z več sto atributi, kjer so vsi atributi pomembni, v praksi redko srečamo, zato si pri takih množicah najprej pomagamo z izbiro podmnožice atributov. Računanje pomembnosti Λ bi bilo smiselno tudi za tako velike množice.

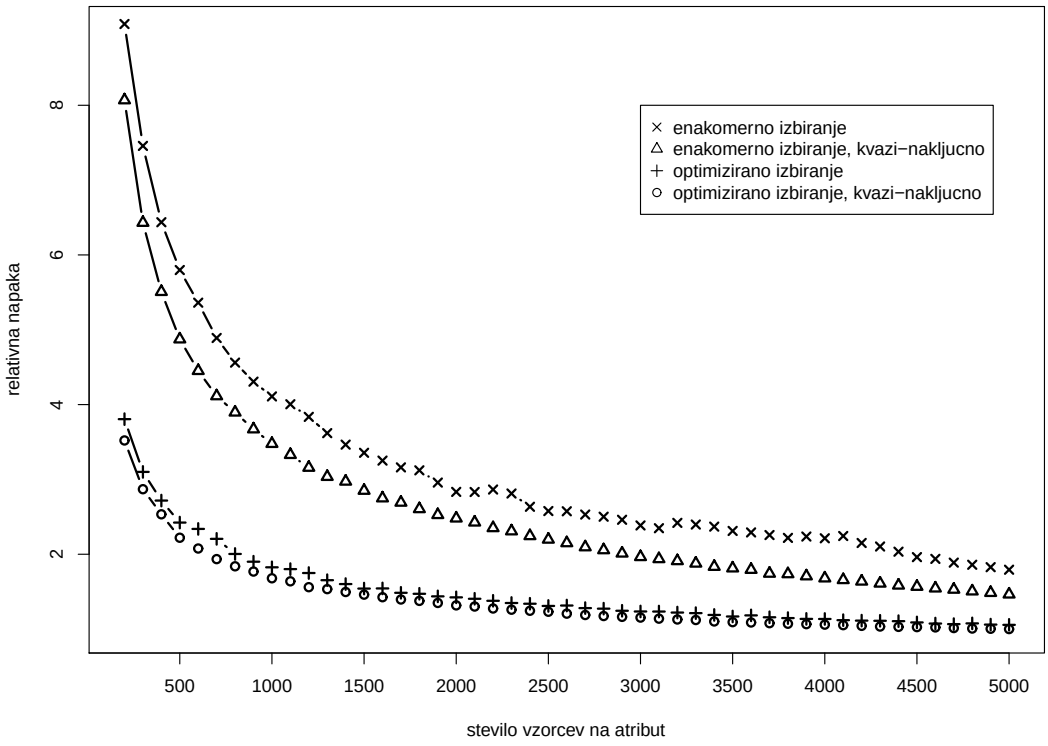
4.2.3 Napaka aproksimacije in optimalna izbira vzorcev

Ker prispevke ne računamo eksaktno, se aproksimirani prispevki razlikujejo od pravih za neko napako. Velikost te napake je v odvisna od števila vzorcev, ki jih vzorčimo za vsak atribut. V povprečju velja, da nam večje število vzorcev zagotovi manjšo napako. Ker smo v praksi običajno omejeni s časom in lahko vzorčimo le določeno število vzorcev, jih želimo med atribute razporediti tako, da čim bolj zmanjšamo pričakovano skupno napako preko prispevkov vseh n atributov. Algoritem, ki nam pove, kako optimalno razporediti vzorce, smo predstavili že v prejšnjem poglavju. V tem podpoglavju empirično določimo njegove lastnosti pri praktični rabi.

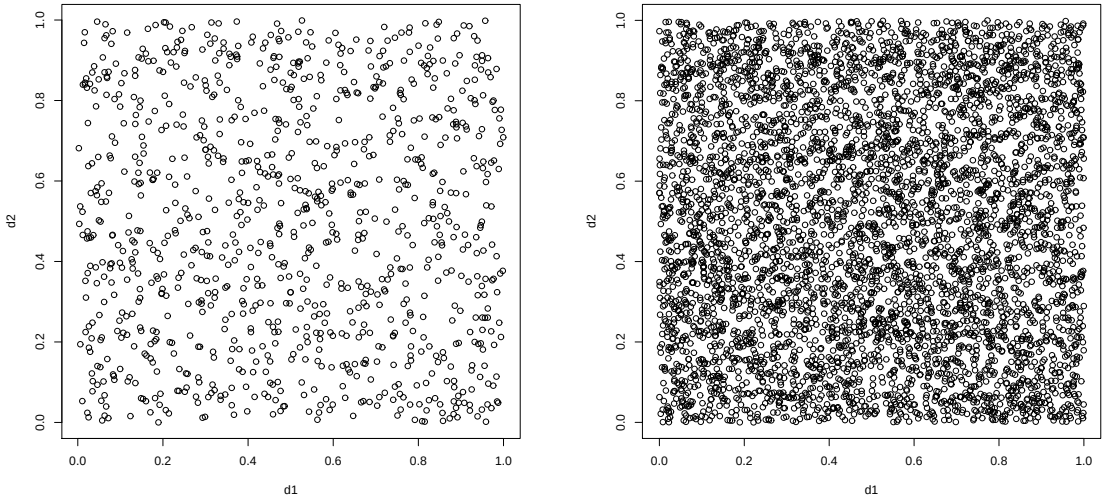
Z vidika izbire vzorcev nas zanimata predvsem dve stvari. Prvič, za koliko se zmanjša napaka aproksimacije, če pri enakem skupnem številu vzorcev namesto enakomerne razporeditve uporabim optimalno razporeditev. In drugič, koliko vzorcev (časa) lahko prihranimo z optimalno izbiro vzorcev, če želimo doseči enako skupno napako kot pri enakomerni razporeditvi. Slika 4.5 prikazuje čas, ki ga pridobimo z optimalno izbiro vzorcev, v odvisnosti od skupnega števila vzorcev.

Ti rezultati predstavljajo povprečje preko vseh parov model/množica. V določenih primerih se lahko zgodi, da z optimalno izbiro vzorcev nič ne pridobimo. Spomnimo se ugotovitve iz podpoglavja 3.5, ki pravi, da je v primeru enakih varianc σ_i^2 enakomerna razporeditev vzorcev optimalna. Sem seveda spadajo tudi primeri, ko se model nič ne nauči in so vsi atributi nepomembni.

V praksi lahko konvergenco dodatno izboljšamo z uporabo zaporedja vzorcev, ki prostor zapolni bolj enakomerno od psevdo-naključnega - tako imenovana kvazi-naključna zaporedja [70, 71]. Za ilustracijo glejte sliko 4.6. Opisani poskus smo ponovili z uporabo Sobolovega kvazi-naključnega zaporedja [41]. Slednje izboljša konvergenco, vendar je pridobitek manjši v primerjavi s pridobitkom, ki ga dosežemo zgolj z optimalno izbiro vzorcev (glej sliko 4.5). Najboljše rezultate dosežemo z uporabo tako optimalne izbire vzorcev kot tudi kvazi-naključnim vzorčenjem.



Slika 4.5: Ker za večino množic podatkov v praksi ni moč izračunati točnih prispevkov, smo za točne prispevke vzeli prispevke, izračunane s 100000 vzorci na atribut. Zabeležene napake so relativne glede na napako pri optimizirani razporeditvi vzorcev in 5000 vzorci na atribut.



(a) psevdo-naključno, 1000 točk

(b) psevdo-naključno, 5000 točk

(c) kvazi-naključno, 1000 točk

(d) kvazi-naključno, 5000 točk

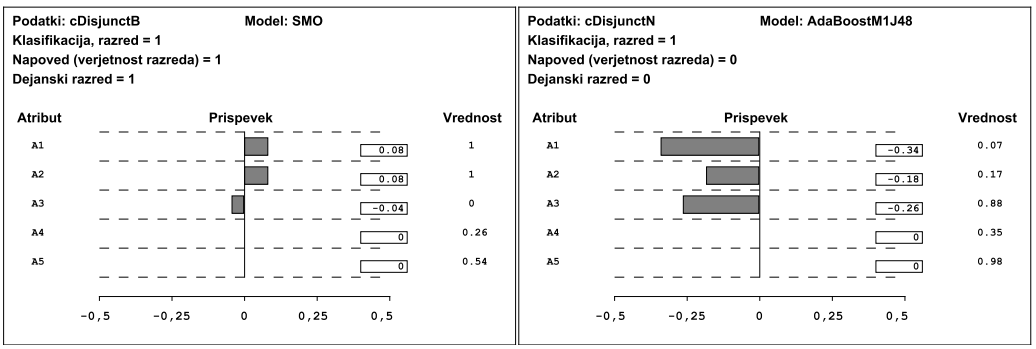
Slika 4.6: Primerjava Sobolovega kvazi-naključnega in psevdo-naključnega zaporedja točk na enotskem kvadratu. Pri slednjem gre za uporabo Javinega generatorja naključnih števil.

4.3 Vizualno preverjanje razlag

V tem podpoglavju ocenimo, če so prispevki atributov razumljivi in v skladu z našim znanjem o modelih in skritih konceptih, ki jih modelirmo. Prešli smo v subjektivno ocenjevanje razlag, pri čemer sami odigramo vlogo strokovnjakov, saj poznamo koncepte, ki se skrivajo za množicami podatkov. Najprej analiziramo razlage posameznih napovedi, nato še razlage modelov.

4.3.1 Razlage napovedi

Razlaga napovedi nekega modela je preprosta vizualizacija prispevkov φ. Slika 4.7(a) je takšna vizualizacija za SMO na množici cDisjunctB in to za napoved, ki smo jo že obravnavali kot ilustrativni primer - disjunkcija $1 \vee 1 \vee 0$. Poleg vrednosti atributov in njihovih prispevkov k napovedi na vizualizaciji najdemo še ime množice podatkov in ime modela, pod njima pa še napoved modela in dejanski izid za ta primer. SMO se koncept dobro nauči in pravilno napove razred 1. Prispevki atributov so podobni tistim, ki smo jih naračunali za optimalni model, kar še potrjuje, da se je SMO pravilno naučil koncepta disjunkcije.

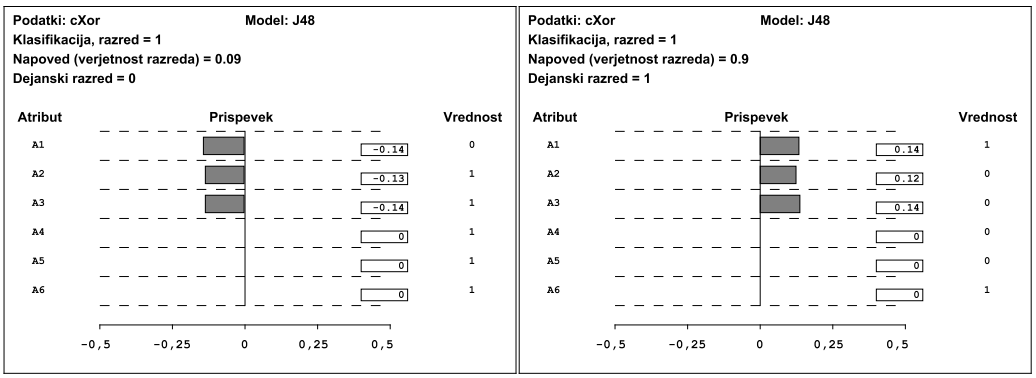


(a) (b)

Slika 4.7: Vizualizaciji prispevkov za primera disjunkcije z diskretnimi (a) in zveznimi (b) atributi.

Tudi za disjunktni koncept z zveznimi atributi (cDisjunctN) metoda pravilno izračuna prispevke. Slika 4.7(b) je vizualizacija prispevkov za model boostinga z odločitvenimi drevesi, ki povsem pravilno napove verjetnost 0 za primer, ko nobeden izmed treh atributov ne izpolnjuje pogoja, ki bi zadostoval za razred 1. Najbolj negativen prispevek pri napovedi ima prvi atribut, sledi tretji, najmanjši negativni prispevek pa ima drugi atribut. To je smiselno, saj je ravno za prvi atribut najbolj verjetno, da bo pogoj izpolnil (0.5), medtem ko je ta verjetnost pri tretjem (0.33) in drugem atributu (0.3) manjša. Prvi atribut je največji "krivec" za razred 0.

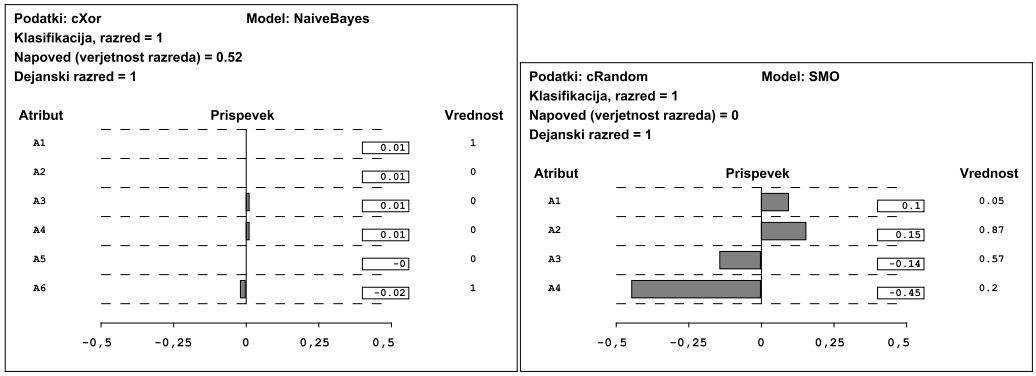
Odločitveno drevo spada med algoritme, ki se dobro naučijo ekskluzivnega ali. Kot pri ilustrativnem primeru iz 3. poglavja, tudi za ta model z metodo izračunamo, da imajo



(a) (b)

Slika 4.8: Vizualizaciji prispevkov za isti model a različna primera iz množice cXor.

vsi trije pomembni atributi enak prispevek. Ko model napove razred 0, so ti prispevki negativni (glej sliko 4.8(a)), ko napove razred 1, pa so pozitivni (glej sliko 4.8(b)). Če se učni algoritem ni zmožen naučiti koncepta ekskluzivnega ali, je to razvidno tudi iz prispevkov (glej primer naivnega Bayesa na sliki 4.9(a)).

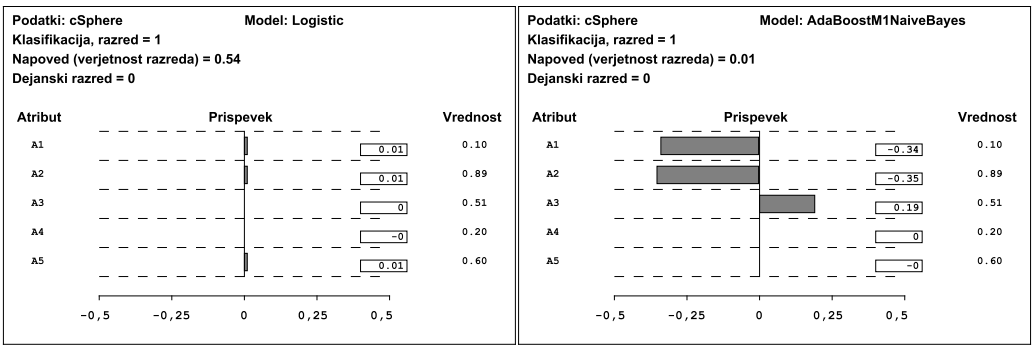


(a) (b)

Slika 4.9: Vizualizaciji prispevkov prikazujeta primer napovedi neuspešnega modela in modela, ki se je preveč prilagodil množici podatkov.

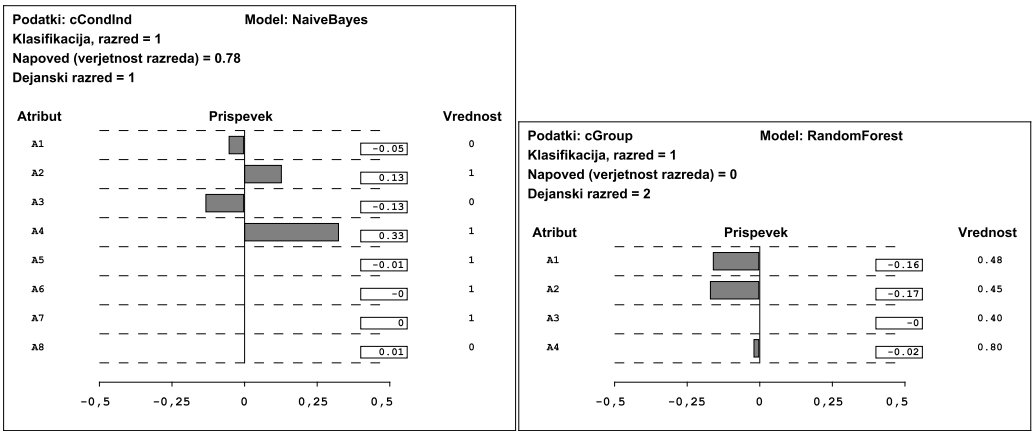
Večkrat smo že omenili, da prispevki odražajo, kar se je model naučil, zato lahko skupaj s točnostjo modela ugotovimo, ali (in kako) se je model preveč prilagodil učnim podatkom. Tako se je zgodilo pri modelu SMO na množici cRandom. Primer napovedi tega modela najdemo na sliki 4.9(b).

Še eno primerjavo napovedi modela, ki se je dobro naučil, in modela, ki se ni, najdemo na slikah 4.10(a) in 4.10(b). Gre za množico cSphere in primer, kjer točka zaradi prvih dveh koordinat leži izven enotski kocki včrtane sfere. Z logistično regresijo tega problema v taki obliki ne moremo modelirati, metaučenje boosting z naivnim Bayesom pravilno napove razred. Iz prispevkov za slednjo napoved razberemo tudi vpliv treh pomembnih atributov.



(a) (b)

Slika 4.10: Vizualizaciji prispevkov za napovedi primerov iz množice cSphere. Model na levi neuspešno modelira koncepte, model na desni uspešno napove primer.

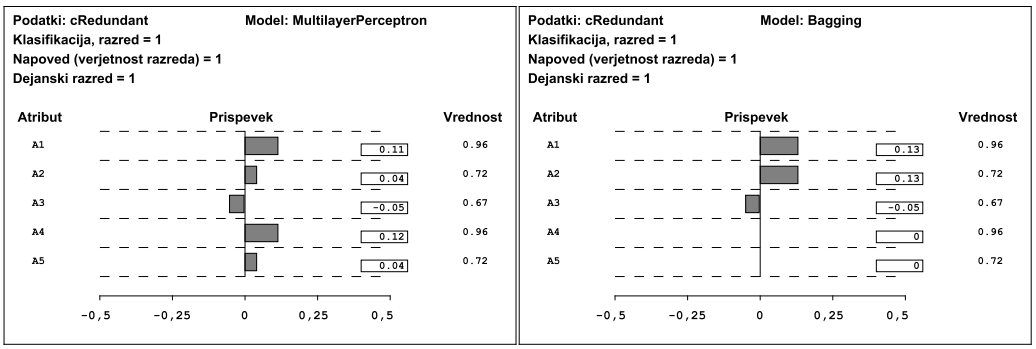


(a) (b)

Slika 4.11: Naivni Bayes oziroma naključni gozdovi sta bila najbolj uspešna na množici cCondInd oziroma cGroup. Prispevki k napovedi so zato v skladu z dejanskim vplivom atributov na razred.

Na sliki 4.11 najdemo še dve vizualizaciji prispevkov. Prvi je za naivnega Bayesa na cCondInd, druga za naključne gozdove na cGroup. V obeh primerih gre za uspešno naučena modela, kar je razvidno tudi iz pravilnih napovedi in prispevkov. Vizualizaciji na sliki 4.12 sta za isti primer iz množice cRedundant, a za različna modela. V tej množici je vseh pet atributov pomembnih, a četrti in peti sta le kopija prvega in drugega atributa. Iz prispevkov je razvidno, da na napoved nevronske mreže vpliva vseh pet atributov, na napoved odločitvenega drevesa pa samo prvi trije. Slednje je posledica delovanja učnega algoritma za gradnjo drevesa, ki najprej izbere eno izmed obeh kopij, druge pa nato ne več, saj v tistem trenutku več ne ponuja koristnih informacij. Učna algoritma na drugačen način obravnavata atribute, a na koncu oba zgradita uspešen model.

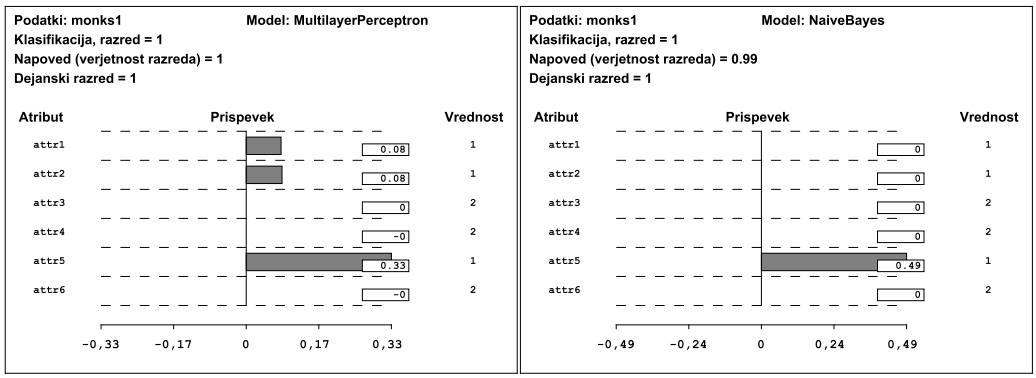
Nadaljujmo z dvema paroma primerov iz množic monks1 in zoo (glej sliki 4.13 in 4.14). Pri monks1 je razred enak 1 natanko takrat, ko imata prva dva atributa enako



(a) (b)

Slika 4.12: Umetna nevronska mreža za razliko od odločitvenega drevesa upošteva tudi kopije atributov pri množici cRedundant. Imamo dva enako uspešna modela, a različni razlagi, saj prispevki odražajo tisto, kar se je model naučil.

vrednost ali pa ima peti atribut vrednost 1. Primer, ko veljata oba pogoja, pravilno napovesta tako nevronska mreža (slika 4.13(a)) kot tudi naivni Bayes (slika 4.13(b)). Slednji napove pravilno, kljub temu, da zaradi predpostavke o pogojni neodvisnosti ne modelira ekvivalence, temveč se zanaša le na drugi pogoj (peti atribut ima vrednost 1). Povedano je razvidno tudi iz vizualiziranih prispevkov.

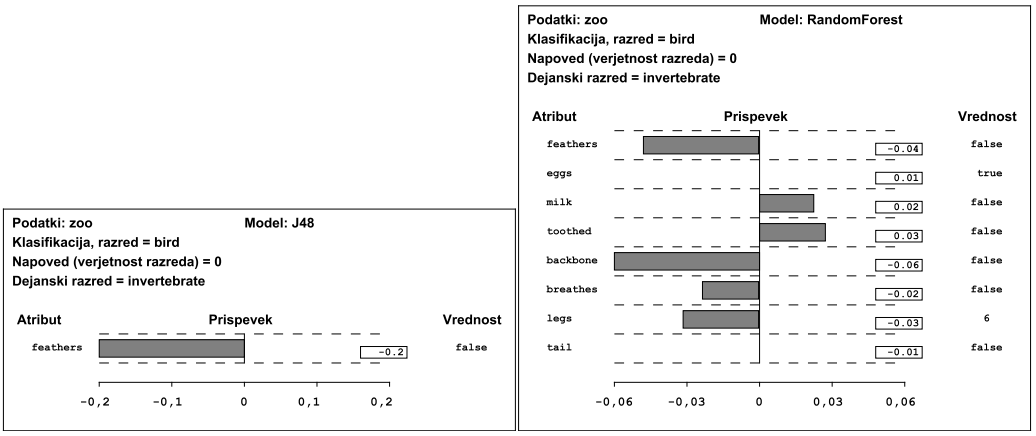


(a) (b)

Slika 4.13: Naivni Bayes ni zmožen modelirati pomembnosti ekvivalence med atributoma, a kljub temu pravilno napove primer iz monks1, saj je dovolj že dejstvo, da je peti atribut enak ena. Umetna nevronska mreža uspešno modelira tudi pomen ekvivalence in se zato bolje obnese na drugih primerih iz te množice.

Drugi par primerov ilustrira razliko med odločitvenimi drevesi in naključnimi gozdovi. Medtem ko je algoritem za gradnjo dreves nagnjen k temu, da upošteva manjše število atributov, naključni gozdovi običajno upoštevajo več atributov, četudi le-ti niso neposredno potrebni za pravilno napoved. Spomnimo se, da smo podobno že opazili na primeru redundantnih atributov (slika 4.11). Na slikah 4.14(a) in 4.14(b) najdemo napovedi teh dveh modelov za isti primer iz množice zoo. Žival, v tem primeru nevretenčar, je opisana

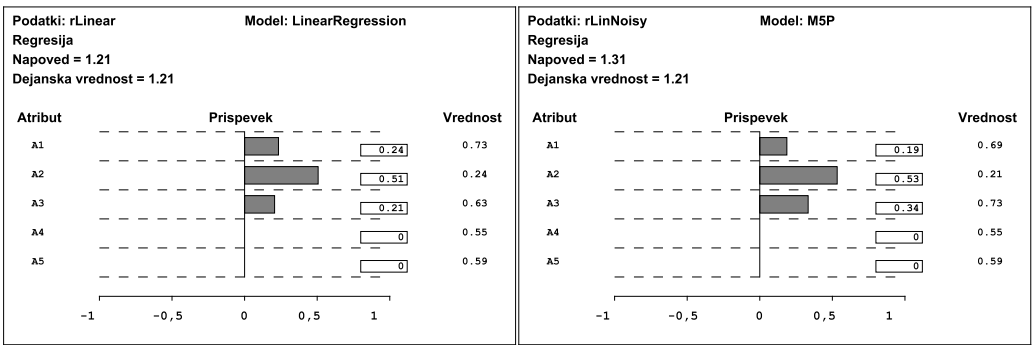
z večjim številom atributov, modela pa napovedujeta verjetnost, da je žival ptica. Iz prispevkov pri odločitvenem drevesu razberemo, da žival ni ptica, saj nima perja. Iz prispevkov pri naključnih gozdovih pa je več pomembnih atributov, ki govorijo proti temu, da je žival ptica: nima hrbtenice, nima perja, ima 6 nog in ne diha. Vrednosti atributov, ki pravita, da žival ne daje mleka in nima zob, pa govorita v prid temu, da je žival ptič.



(a) (b)

Slika 4.14: Medtem ko Naključni gozdovi pri svoji odločitvi upoštevajo večje število atributov, je za odločitveno drevo že odsotnost perja zadosten dokaz, da žival ni ptica.

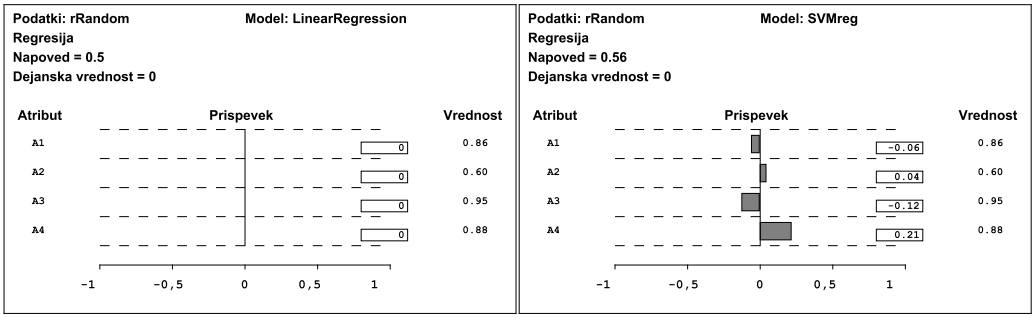
Sedaj si poglejmo nekaj regresijskih problemov. Najbolj preprost netrivialen regresijski problem je linearni cLinear, ki ga dobro opišejo vsi modeli. Poglejmo si vizualizaciji prispevkov za primer iz te množice (glej sliko 4.15(a)) in isti primer s šumnimi vrednostmi pomembnih atributov (glej sliko 4.15(b)). Pozornejše bralce bo za hip morda zmotilo, da ima drugi atribut pozitiven prispevek pri vrednosti 0.24, saj ima vendarle negativen koeficient -2. Spomnimo se, da je v tem kontekstu njegova pričakovana vrednost $-2 \cdot 0.5 = -1$, kar je za 0.52 več, kot bi pričakovali. Približno tak je tudi njegov prispevek.



(a) (b)

Slika 4.15: Na preprostem linearnem problemu rLinear so vsi modeli uspešni. Prispevki nam za primera na sliki povedo, da imajo vrednosti atributov pozitiven vpliv na napoved.

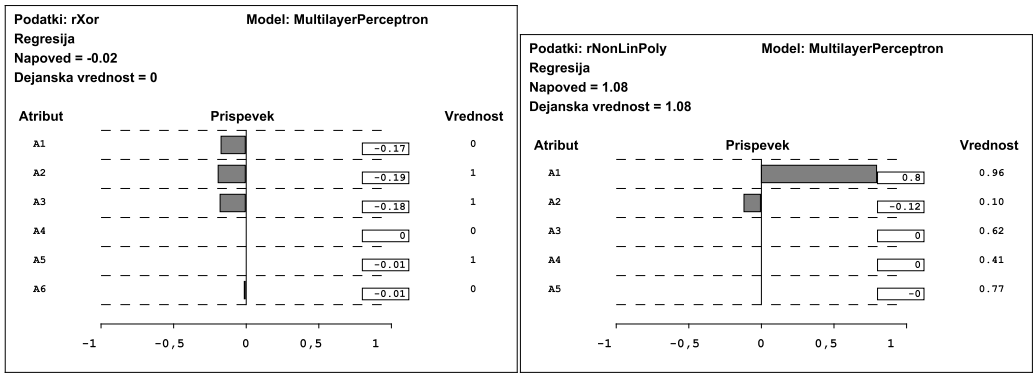
Tudi regresijskim množicam podatkov se lahko modeli preveč prilagodijo. Regresijski različici rRandom se zopet preveč prilagodi regresijski SVM (glej sliko 4.16(b)). Linearna regresija, ki je robustna ko gre za preveliko prilagajanje, pa se (pravilno) ne nauči ničesar (glej sliko 4.16(a)).



(a) (b)

Slika 4.16: Regresijski SVM se preveč prilagodi množici rRandom, ki ne vsebuje nobenih konceptov (razred ni povezan z atributi). Linearna regresija je bolj robustna in se (pravilno) ne nauči ničesar.

Podobno kot pri klasifikaciji, je tudi pri regresijski različici rXor razviden prispevek pomembnih atributov, če se le model pravilno nauči koncepta (slika 4.17(a)). Tudi pri drugih nelinearnih konceptih, kot je na primer polinom tretje stopnje, metoda smiselno razdeli prispevke (glej sliko 4.17(b)).

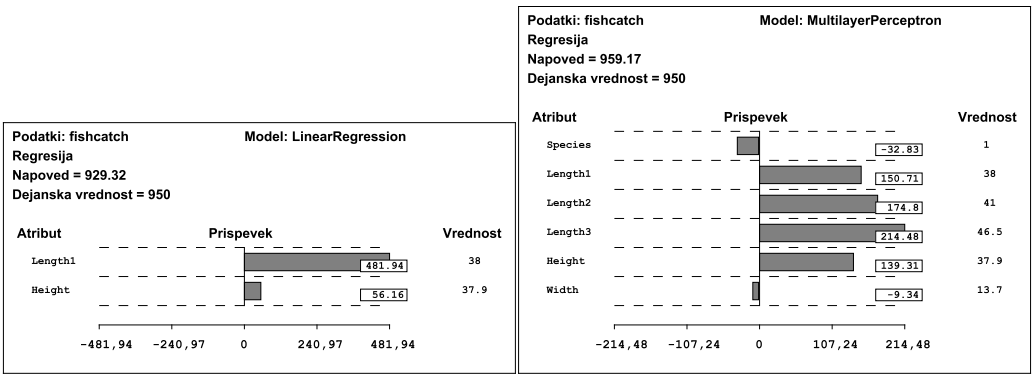


(a) (b)

Slika 4.17: Tudi pri regresijskih množicah in nelinearnih problemih izračunani prispevki odražajo vpliv atributov na model.

Za uporabnike so najbolj zanimive vizualizacije prispevkov za množice, ki temeljijo na realnih primerih. Pozor, pri nekaterih vizualizacijah, ki sledijo, smo odstranili atribute z zanemarljivo majhnimi prispevki in s tem zmanjšali njihovo velikost!

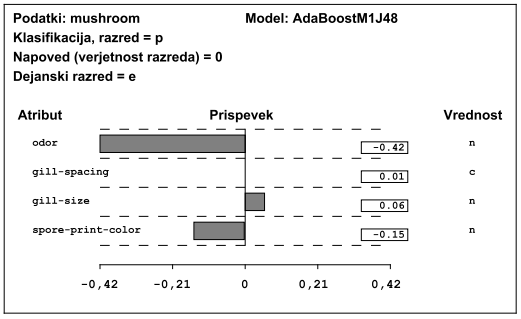
Prva primera sta iz množice podatkov fishcatch (glej sliki 4.18(a) in 4.18(b)). Množica je sestavljena iz ujetih rib in njihovih dimenzij, napovedujemo pa težo ribe. Najmanj



(a) (b)

Slika 4.18: Linearna regresija pri napovedovanju teže ribe upošteva le njeno dolžino od nosa do začetka repa. Nevronska mreža upošteva tudi druge dolžine ter vrsto ribe, zato je v povprečju bolj uspešna pri napovedovanju.

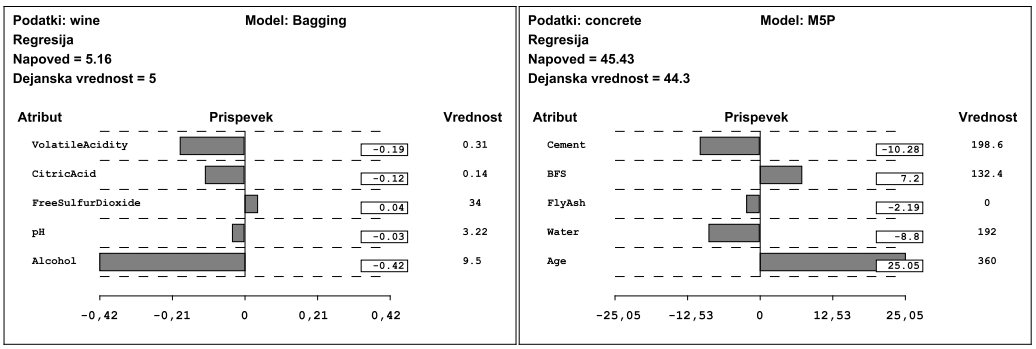
uspešen model na tej množici je linearna regresija, najbolj pa umetna nevronska mreža. Na podlagi tega lahko sklepamo, da ne gre za povsem linearen problem. Prej omenjeni vizualizaciji pokažeta, da oba modela podani primer napovesta približno enako dobro, a nevronska mreža pri tem upošteva vrsto ribe, vse tri dolžine (od nosa do začetka repa, zareze na repu, konca repa) in višino, medtem ko linearna regresija le dolžino od nosa do začetka repa.



Slika 4.19: Sodeč po prispevkih atributov za primer neke gobe iz množice mushroom in model boostinga z odločitvenimi drevesi, lahko gobo brez skrbi pojemo. Goba nima vonja, odtis spor pa je rjav.

Naslednji primer je iz klasifikacijske množice podatkov mushroom, ki opisuje različne vrste gob, za katere nas zanima, ali so užitne ali strupene. Boosting z odločitvenimi drevesi je kot večina modelov dosegel maksimalno točnost, zato pravilno napove tudi primer na sliki 4.19. Model napove, da goba zagotovo ni strupena, prispevki pa pravijo, da k temu najbolj prispeva to, da goba nima vonja. Proti strupenosti govori tudi rjava barva odtisa spor, ki prav tako velja za pomemben diagnostični indikator pri določanju vrste gob.

Na množici wine je najbolj uspešno metaučenje bagging z regresijskim drevesom. Množica je sestavljena iz različnih vin, napovedujemo pa oceno kvalitete vina, ki so jo strokov-



(a) (b)

Slika 4.20: Na teh dveh primerih metaučenje bagging z odločitvenimi drevesi in regresijsko drevo napovedujeta kvaliteto vina in trdnost cementa. Na slabšo oceno vina vplivata vsebnost alkohola ter, nekoliko manj, ocetna kislost in citrusna kislina. Trdnost betona pa je višja zaradi njegove starosti, a nižja kot bi bila lahko, zaradi količine cementa in vode v mešanici.

njaki pripisali posameznemu vinu. Model približno dobro napove, da bo vino iz primera na sliki 4.20(a) dobilo podpovprečno oceno 5. Vizalizacija prispevkov nam pomaga razumeti razloge. Glavni razlog je vsebnost alkohola, nekoliko manj pomembna pa količina ocetne kislosti in citronske kisline. Strokovnjaki za vino pravijo, da povečanje vsebnosti alkohola, ki znaša "le" 9.5%, kvečjemu izboljša kakovost vina [19].

Zaključimo s "konkretnim" primerom napovedovanja moči betona na podlagi sestavin mešanice in starosti. Za primer na sliki 4.20(b) regresijsko drevo (drugi najuspešnejši model na tej množici) skoraj točno napove moč cementa. Prispevki so si nasprotujoči. Starost 360 dni govori v prid večji moči betona, količina cementa in vode pa proti. Strokovnjaki potrjujejo, da moč betona raste s starostjo, medtem ko je razmerje skoraj 1:1 med cementom in vodo neprimerno, saj delež vode v praksi le redko preseže 40%.

Če povzamemo, so vizalizacije prispevkov uporabno orodje za razlago posameznih napovedi. Kljub temu v praksi pogosto potrebujemo tudi širšo sliko delovanja modela in ne zgolj prispevke pri napovedi. Vzemimo za primer sliko 4.20(a), ki predstavlja napoved kvalitete nekega vina. Iz prispevkov lahko razberemo, da je količina algohola v tem primeru slaba za kvaliteto vina. Vendar, če ne poznamo strokovnega ozadja, ne moremo sklepati, kako bi tak problem odpravili – s povečanjem ali zmanjšanjem količine alkohola? V pomoč so nam razlage modelov, katerih opis sledi.

4.3.2 Razlage modelov

Za uvod začnimo z eno izmed najbolj preprostih množic, ki jo vsi modeli zelo dobro modelirajo – rLinear. Na sliki 4.21(a) najdemo razlago modela linearne regresije za množico rLinear. Slika ni nič drugega kot vizualizacija splošnega prispevka posameznih vrednosti atributa, za vsak atribut ($\psi_i(x)$, glej tudi slike 2.7(a) in (b),2.8(a) in (b) in 2.9). Na ab-

cisi najdemo vrednosti atributov (v primeru diskretnih atributov vrednosti najdemo poleg križca, ki označuje prispevek te vrednosti – glej npr. sliko 4.22(b)), na ordinati pa velikost prispevka. Poleg splošnih prispevkov vrednosti smo v vizualizacijo pri vsakem atributu vključili še njegovo pomembnost $\sqrt{\mathrm{Var}[\Lambda_i]}$, katere vrednost je označena z debelejšo sivo črto. Pozor, pomembnost smo korenili zato, da dobimo enoto, ki ustreza ordinati, in tako poenostavimo vizualizacijo. Pomembnost atributa je še posebej pomembna v primerih, ko je povprečni prispevek vrednosti atributa enak nič, atribut pa je vseeno pomemben (glej sliko B.8).

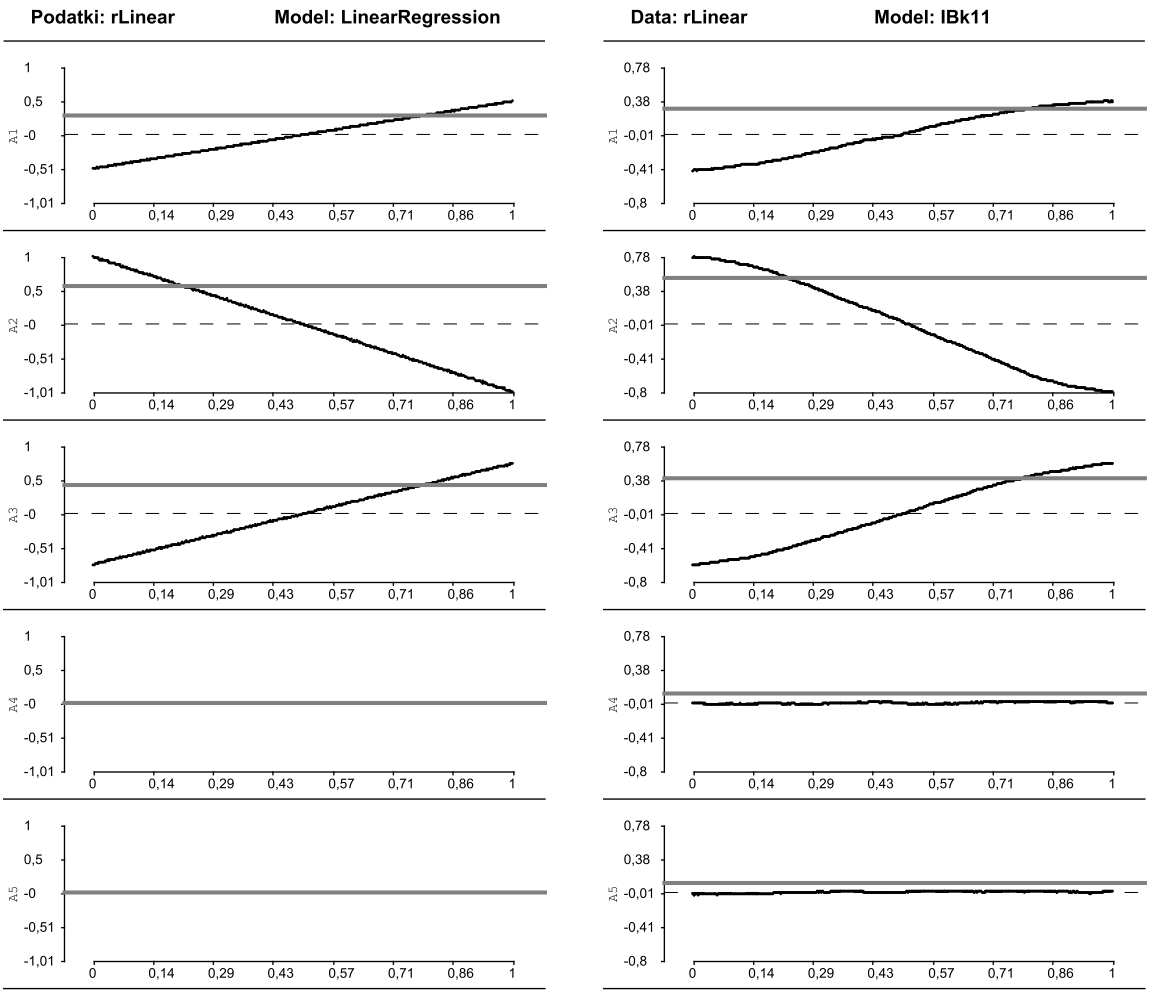
Če se vrnemo k sliki 4.21(a) vidimo, da je linearni model dobro opisal tri pomembne atribute, nepomembnih atributov pa ne upošteva. Iz črt, ki označujejo pomembnost atributov lahko razberemo, da je najpomembnejši drug, nato tretji, na koncu pa še prvi atribut, kar je v skladu s koeficienti in dejstvom, da smo razlagali v kontekstu enako verjetnih primerov.

Ena od prednosti razlage je tudi v tem, da lahko modele primerjamo med seboj. Na sliki 4.21(b) je vizualizacija za metodo najbližjih sosedov za isto množico. Primerjava z linearno regresijo nam razkrije, da metoda najbližjih sosedov upošteva, sicer v manjši meri, tudi nepomembna atributa. Oblike funkcij splošnih prispevkov vrednosti so podobne, a opazimo, da so pri metodi najbližjih sosedov le-te nekoliko ukrivljene na obeh koncih. Če si malo bolje predstavljamo delovanje te metode, lahko hitro ugotovimo razlog. Ko se bližamo robu prostora atributov, je na strani, ki je bližje robu, vedno manj primerov, kot na strani, ki je bližje središču prostora. V povprečju na robu vzamemo več primerov z manj ekstremnimi vrednostmi, kar ukrivi funkcijo.

Sedaj, ko smo spoznali razlage primerov in modelov, se lahko vrnemo na uvodni primer s profesorji in študenti. Zvezo med obema atributoma in možnostmi študenta, da izpit opravi, smo modelirali z odločitvenim drevesom in izračunali prispevke v različnih kontekstih (množice testA, testB in testC). Vizualizacije prispevkov za študenta, ki se ni učil ter bo vprašan, najdemo na sliki 4.23.

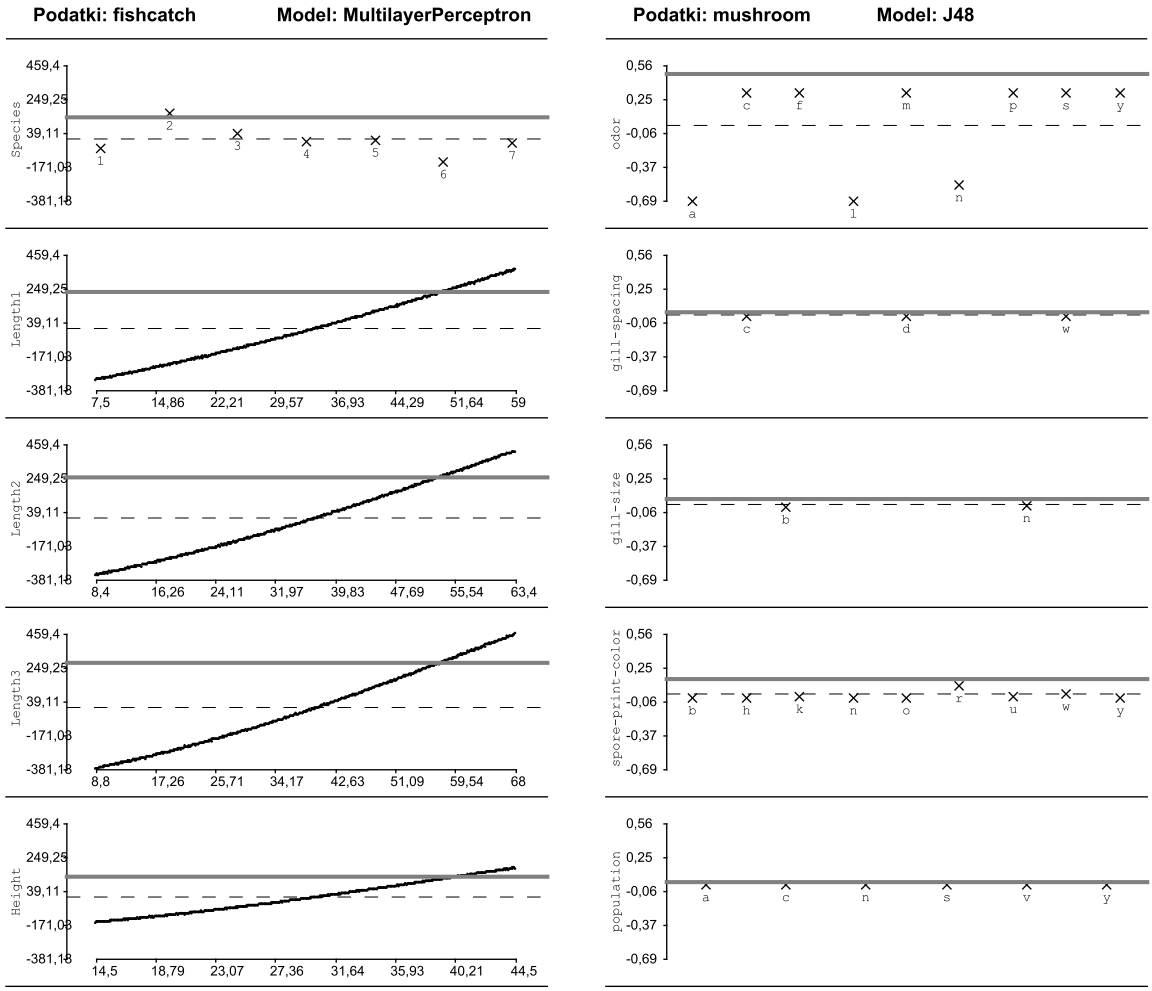
Že pri paru primerov na sliki 4.18 smo povedali, da nevronska mreža pri napovedi teže ribe upošteva večje število atributov. Natančneje, za napovedi umetne nevronske mreže so pomembni vsi atributi, v manjši meri le širina in spol ribe. Več informacij o splošnih prispevkih vrednosti posameznih atributov najdemo na sliki 4.22(a). Na sliki 4.22(b) je vizualizacija petih najpomembnejših atributov za odločitveno drevo pri napovedovanju užitnosti gobe. Kot smo sklepali že po primeru na sliki 4.19, je najpomembnejši indikator ravno vonj gobe. Določen pomen ima tudi barva odtisa spor, ostali atributi pa so za napovedi tega modela skoraj nepomembni.

Zaradi boljše preglednosti smo preostale razlage modelov opisali v dodatku. Bralci, ki jih zanimajo dodatni ilustrativni primeri uporabe metode za razlago modelov, naj pred nadlednjim poglavjem preberejo dodatek B.



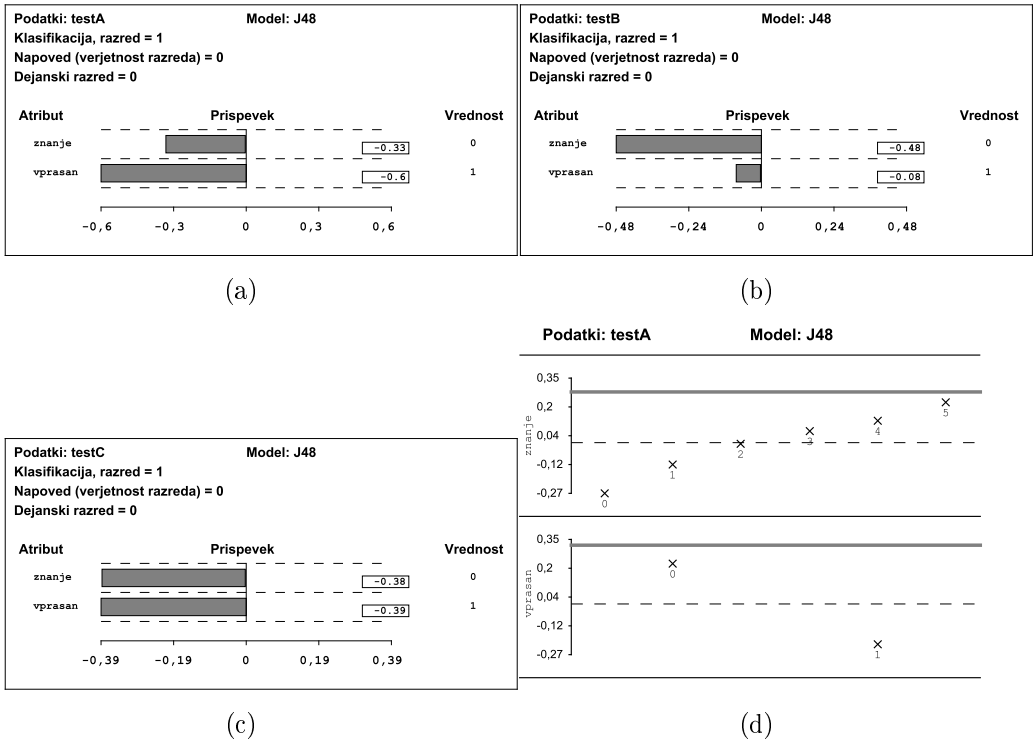
(a) (b)

Slika 4.21: Tako linearna regresija kot tudi metoda najbližjih sosedov (k=11) uspešno modelirata linearni problem rLinear, čeprav z manjšimi razlikami.



(a) (b)

Slika 4.22: Na levi strani je vizualizacija modela umetne nevronske mreže, ki pri napovedovanju teže ribe uporablja vse tri dolžine, višino in vrsto ribe. Na desni strani najdemo prispevke najpomembnejših atributov za model J48 pri napovedovanju užitnosti gobe.



Slika 4.23: Velikost prispevkov za študenta, ki se ni učil, a bo vprašan, se spreminja glede na uporabljeni kontekst. Ko profesor redkeje sprašuje (a), je bolj "krivo" to, da je bil vprašan. Ko je spraševanje pogosto (b), je krivo pomanjkanje učenja. Pri enako verjetnih primerih (c) sta obe vrednosti enako pomembni. Iz razlage modela (d) je razviden tudi vpliv posameznih vrednosti obeh atributov.

Poglavje 5

Posebni primeri uporabe in praktična raba

V tem poglavju opišemo aplikacijo metode na realni problem – napovedovanje ponovitve raka dojke, študijo uporabnosti razlage za običajne uporabnike, in primer uporabe metode za izbiro podmnožice atributov. Omeniti velja tudi uporabo predlagane metode v raziskavah drugih avtorjev. Teh študij podrobneje ne opišemo, temveč jih le naštejemo. V eni izmed teh študij so se avtorji ukvarjali z napovedovanjem največje strižne sile na steni žile iz geometrije žile in podatkov pridobljenih iz simulacij hemodinamike [74]. Splošni prispevki vrednosti atributov so bili uporabljeni kot orodje za vizualni pregled vpliva atributov na umetno nevronsko mrežo in izbiro podmnožice atributov, ki je poenostavila model in izboljšala njegovo točnost. Če uporaljamo klasifikator PRBF (Probabilistic Radial Basis Function network), je metodo moč prilagoditi za bolj učinkovito rabo [80]. Metoda je bila uporabljena tudi za vizualizacijo in razlago modelov za napovedovanje bolezni srca in ožilja [55] in modelov za napovedovanje poslovnih izidov podjetja iz meril kvalitete organizacije združbe [73].

5.1 Napovedovanje ponovitve raka dojke

Napovedovanje ponovive raka dojke je pomemben problem v medicinski prognostiki. Kot pri večini prognostičnih problemov, tudi tu dosežemo vsaj tako dobre rezultate kot strokovnjaki (omejeni na uporabo istih faktorjev), če je le na voljo dovolj podatkov [51]. Kljub temu si zdravniki pogosto neradi pomagajo s takimi napovedmi, predvsem zaradi pomanjkanja zaupanja. Tak odnos zdravnikov je razumljiv, če vzamemo v obzir pomembnost teh napovedi in tveganje, ki je povezano z napačnimi odločitvami. Ravno premoščanje te vrzeli med napovedmi modelov in uporabnikom je glavni praktični cilj razlage. Napovedi želimo razložiti na način, ki bi zdravnikom omogočil, da lažje povežejo svoje znanje z delovanjem modela.

V letih 2008 in 2009 smo dobili priložnost, da predlagano metodo razlage preverimo

Tabela 5.1: Opis atributov.

Ime atributa	Opis atributa
menop	binarni atribut, ki označuje menopavza
stage	stadij; 1: manj kot 20mm, 2: med 20mm in 50mm, 3: več kot 50mm
grade	gradus; 1: dobro, 2: srednje, 3: slabo, 4: se ne uporablja, 9: ni določen
histType	histološki tip; 1: duktalni, 2: lobularni, 3: drugo
PgR	Receptor progesterona (v fmol na mg proteina); 0: manj kot 10, 1: več kot 10, 9: neznan
invasive	invazivnost; 0: brez, 1: koža, 2: prsna bradavica, 3: prsna bradavica in koža, 4: stena ali mišica
nLymph	število pozitivnih bezgavk; 0: 0, 1: med 1 in 3, 2: med 4 in 9, 3: 10 ali več
famHist	anamneza; 0: ni rakavih obolenj, 1: rak dojke, jajčnikov ali prostate v prvi generaciji, 2: rak dojke, jajčnikov ali prostate v drugi generaciji, 3: neznan ginekološki rak, 4: rak črevesja ali slinavke 5: druga neznana rakava obolenja, 9: ni določen
LVI	binarni atribut, ki označuje prisotnost limfovaskularne invazija
ER	Receptor estrogena (v fmol na mg proteina); 1: manj kot 5, 2: 5 do 10, 3: 10 do 30, 4: več kot 30, 9: ni določen
maxNode	premer največje bezgavke; 1: manj kot 15mm, 2: med 15 in 20mm, 3: več kot 20mm
posRatio	delež pozitivnih bezgavk med vsemi odstranjenimi; 1: 0, 2: manj 10%, 3: med 10% in 30%, 4: več kot 30%
age	starostna skupina; 1: manj kot 40, 2: 40-50, 3: 50-60, 4: 60-70, 5: več kot 70 let

na medicinskih podatkih. V sodelovanju z Onkološkim Inštitutom Ljubljana smo analizirali množico podatkov o 1035 pacientkah, ki so obolele za rakom dojke. Vsaka pacientka je predstavljena s primerom, atributi pa ustrezajo vrednostim, ki so bile izmerjene pred in takoj po operaciji dojke. Pacientke so spremljali še več let po operaciji in zabeležili čas morebitne ponovitve raka dojke oz. čas zadnjega kontrolnega obiska. Z odstranitvijo pacientk, ki so bile spremljane manj kot 10 let po operaciji (teh je bilo 154), smo množico preoblikovali v obliko, ki je bila primerna za napovedovanje, ali se bo bolezen ponovila v desetih letih ali ne (glej tabelo 5.1). Podrobnejše rezultate te študije, ki je poleg razlage vključevala tudi analizo zanesljivosti napovedi, najdemo v [94]. V tem podpoglavju povzamemo najpomembnejše rezultate, povezane z razlago napovedi.

Kvaliteto napovedi različnih modelov in dveh onkologov smo primerjali na neodvisni množici 100 testnih primerov. Čeprav razlike niso dovolj velike, da bi sklepali, da so bili modeli statistično značilno boljši od onkologov, pa lahko sprejmemo hipotezo, da niso slabši od onkologov. Najboljši rezultat je dosegel naivni Bayes, ki je že sam po sebi

transparenten, zato lahko dobimo razlago napovedi, s prispevki. Ta oblika razlage je zdravnikom všeč, saj ustreza njihovemu lastnemu načinu razmišljanja [51]. V 3. poglavju smo pokazali, da metoda, ki smo jo predlagali, za naivni Bayes deluje enako kot zanj specifična razlaga. V tem primeru predlagana metoda uspešno nadomesti specifično metodo. Breimanovi naključni gozdovi [13] so bili prav tako boljši od onkologov. Napovedi tega modela so manj transparentne od napovedi naivnega Bayesa, zato je bila to priložnost, da preverimo metodo razlage v praksi. Med štirimi podmnožicami ni statistično značilnih razlik v številu nestrinjanj.

Poskus je potekal na sledeč način. Sto testnih primerov smo razdelili na štiri podmnožice, glede na razred (ponovitev/ ni ponovitve) in napoved modela (pravilna napoved / napačna napoved). Želeli smo preveriti, da razlaga deluje za oba razreda, v primerih, ko se model zmoti, in v primerih, ko napove pravilno. Zelo pomembno je, da se zdravnik strinja z razlago tudi v primerih, ko se je model zmotil. Iz vsake podmnožice smo nato naključno izbrali pet primerov ter izračunali prispevke atributov k napovedi. Vizualizacije prispevkov je ocenil onkolog, pri čemer je za vsak prispevek določil, če se strinja tako s smerjo prispevka kot z njegovo velikostjo (glej sliko 5.1). Izmed 13 atributov jih onkologi v vsakdanji praksi uporabljajo le 9. Prispevkov atributov histType, famHis, maxNode in posRatio nismo ocenjevali. Skupno smo dobili $9 \times 20 = 180$ ocen prispevkov, ki so predstavljene v tabeli 5.2.

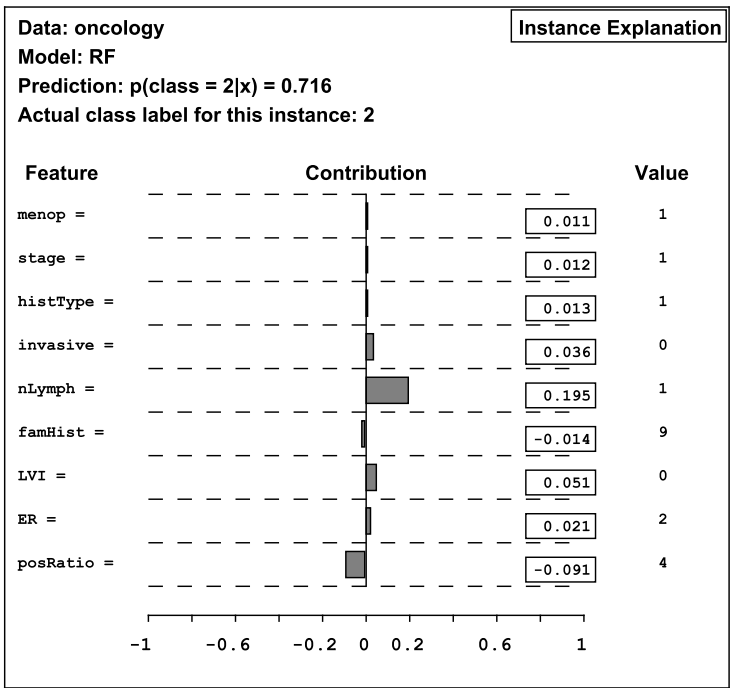
Naknadna analiza teh rezultatov, ki smo jo opravili skupaj z onkologi, je prinesla nekaj dodatnih ugotovitev. Enajst od enaindvajsetih nestrinjanj je bilo bodisi pri atributu *age*, ki opisuje starost, bodisi pri atributu *menop*, ki je prav tako povezan s pacientkino starostjo. Prispevki, ki jih je ponudila naša metoda za razlago, so kazali, da mladost govori proti ponovitvi. To je v nasprotju s konsenzom med onkologi, da mladost govori v prid ponovitvi. Ponoven pregled osnovnih učnih podatkov je pokazal, da je napaka posledica pristranskih podatkov. Onkologi so pojasnili, da je bilo v podatkih, ki smo jih imeli na voljo, nenavadno velik delež mladih pacientk, ki so bile zdravljene z terapijo, kar je posledično zmanjšalo možnost ponovitve. Model se je pravilno naučil podatkov in metoda je pravilno razložila model. Četudi bi predpostavili, da je preosalih 10 nestrinjanj posledica pomanjkljivosti razlage, so se onkologi strinjali z 170 od 180 ($\approx 95\%$) prispevkov.

5.2 Preverjanje splošne uporabnosti razlage s prispevki

V 2. poglavju smo opisali veliko različnih pristopov k razlagi modelov za napovedovanje. Pri ovrednotenju teh pristopov v praksi gre v večini primerov za aplikacije na nekem področju (medicina, biologija, finance, marketing,...), kjer nato manjše število strokovnjakov iz področja (subjektivno) ovrednoti uporabnost predlagane metode za razlago. Podobno smo v sodelovanju z onkologi preverili uporabnost naše metode, kar smo opisali v prejšnjem razdelku. Seveda je takšno preverjanje smiselno, saj so navsezadnje ravno ti strokovnjaki najbolj verjetni uporabniki metode in na podlagi njihovega pozitivnega mne-

Tabela 5.2: Prave in napovedane vrednosti za 20 primerov. Z ✖ so označeni atributi, pri katerih se onkolog ni strinjal s prispevkom, ki ga je ponutila metoda razlage.

	Razred		Atributi								
Št.	Dejanski	Napoved	age	ER	LVI	nLy.	inv.	PgR	grade	stage	menop
1	2	2	✔	✔	✔	✔	✔	✔	✔	✔	✖
2	2	1	✔	✔	✔	✔	✔	✔	✔	✔	✔
3	2	2	✔	✔	✔	✔	✔	✔	✖	✔	✔
4	2	1	✖	✔	✔	✔	✔	✔	✔	✔	✖
5	2	1	✔	✔	✔	✔	✔	✔	✔	✔	✔
6	2	2	✖	✔	✔	✔	✔	✔	✔	✔	✖
7	2	1	✖	✖	✔	✖	✖	✔	✔	✔	✔
8	2	2	✖	✔	✔	✔	✔	✔	✔	✔	✔
9	2	2	✔	✔	✔	✔	✔	✔	✔	✔	✔
10	2	1	✔	✔	✔	✔	✔	✔	✖	✔	✔
11	1	1	✔	✔	✔	✔	✔	✔	✔	✔	✔
12	1	1	✔	✖	✔	✖	✔	✔	✖	✔	✔
13	1	2	✔	✔	✔	✔	✔	✔	✔	✔	✔
14	1	2	✖	✔	✔	✔	✔	✔	✔	✔	✔
15	1	1	✔	✔	✔	✔	✔	✔	✔	✔	✔
16	1	2	✔	✔	✔	✔	✔	✔	✔	✔	✔
17	1	2	✖	✔	✔	✔	✔	✖	✖	✔	✖
18	1	1	✖	✔	✔	✔	✔	✔	✔	✔	✔
19	1	1	✔	✔	✔	✔	✔	✔	✔	✔	✔
20	1	2	✔	✔	✔	✔	✔	✔	✔	✔	✔
Napake			7	2	0	2	1	1	4	0	4



Slika 5.1: Vizualizacija prispevkov atributov k napovedi ponovitve raka za pacientko s podanimi vrednostmi atributov. Model, ustvarjen z algoritmom naključni gozdovi, je napovedal verjetnost 0.284 ponovitve raka v naslednjih 10 letih. Napoved je bolj ugodna od povprečja. Razlaga pravi, da pacientki najbolj v prid govori majhno število pozitivnih bezgavk, pripomore pa tudi odsotnost limfno-vaskularne invazije. Najbolj negativno pa na napoved vpliva visok delež pozitivnih bezgavk med vsemi odstranjenimi.

nja lahko do neke mere sklepamo, da bo metoda uporabna tudi na drugih področjih. Po drugi strani pa je uporabnost razlage smiselno tudi empirično preveriti na večjem številu uporabnikov.

Študij, v katerih bi merili neposredne ali posredne učinke razlage modelov na uporabnike, je malo in se osredotočajo na odločitvena pravila in odločitvene tabele. Huysmans in sod. [40] so v študiji, v katero je bilo vključenih 51 podiplomskih študentov, primerjali uporabnost odločitvenih tabel, binarnih odločitvenih dreves in pravil. Poskus je bil sestavljen iz različnih nalog, povezanih z ocenjevanjem kreditne sposobnosti, pri čemer velja poudariti, da uporabniki niso imeli preteklih izkušenj s to domeno ali z omenjenimi predstavitvami znanja. Uporabnikom so zastavili več nalog in merili točnost napovedi, hitrost napovedi in uporabnikovo zaupanje v lastne napovedi. Zaključili so, da so odločitvene tabele najboljše in pri uporabnikih najbolj priljubljene. Omeniti velja še [57], kjer so se osredotočili na odločitvena drevesa, in delo [2], kjer so se odsredotočili na odločitvena drevesa in odločitvena pravila.

Odločili smo se, učinke razlage primerov s prispevki posameznih atributov preverimo s praktičnim poskusom, podobnim tistemu v [40], ki bo vključeval večje število splošnih uporabnikov. Vsakemu uporabniku naj bi ponudili več primerov iz nekega koncepta, na

podlagi katerih bo poskušal razbrati povezave med atributi in ciljno spremenljivko ter napovedati nekaj novih primerov, za katere vrednost ciljne spremenljivke ni znana.

Naša glavna hipteza je bila, da primeri napovedi, katerim dodamo prispevke posameznih atributov, izboljšajo razumevanje modela oz. konceptov, ki povezujejo vhodne spremenljivke (atribute) in ciljno spremenljivko. Nivo razumevanja konceptov je tesno povezan z uporabnokovo zmožnostjo napovedovanja vrednosti ciljne spremenljivke za nove in neznane primere, zato smo si merjenje razlike v razumevanju zastavili kot merjenje razlike v točnosti napovedi.

Osnova za testiranje sta dva vprašalnika, ki ju najdete v dodatku C, na slikah C.1 in C.2. Od tu dalje testa poimenujemo C1 in C2, pri čemer C1-brez pomeni test C1, pri katerem sta razlaga in opis razlage izpuščena. Naš namen je bil testiranje splošnih uporabnikov, zato koncepta C1 in C2 vsebujeta splošno znane pojme. Pri C1 napovedujemo frekvenco oglašanja murna, ki je premo-sorazmerna temperaturi, pri C2 pa delež uničenih insektov, ki je odvisen od izbire insekticida in količine inekticida, ki je z deležem uničenih insektov nelinearno povezana. Tistim, ki so reševali teste, te povezave seveda niso bile razložene. Prebrana so bila le kratka navodila, ki so pojasnila način in čas reševanja.

5.2.1 Poskus št. 1

Najprej navedimo rezultate poskusa, v katerem je sodelovalo 56 študentov prvega letnika Fakultete za Računalništvo in informatiko. Predpostavili smo, da gre za populacijo, ki nima izkušenj z odkrivanjem zakonitosti v podatkih ali razlago v obliki prispevkov posameznih atributov. Vsak izmed študentov je prejel en list, ki je na prvi strani vseboval test brez razlage, na drugi pa test z razlago. Polovica (28) študentov je prejela list C1-brez / C1, polovica pa list C2-brez / C2.

Vsak študent je imel na voljo 8 minut časa za reševanje prve in 8 minut časa za reševanje druge strani. Izbira časa reševanja je bila motivirana z željo po izločitvi vpliva časovne stiske. V predhodnem poskusu z desetimi študenti, ki jim nismo postavili omenjene časovne omejitve, je bil najdaljši izmerjen čas reševanja dobrih sedem minut.

Za vsako vprašanje posebej smo vzorce rangirali glede na srednjo kvadratno napako napovedi. Z drugimi besedami, ubrali smo neparametrični pristop, s čimer smo se izognili vplivu osamelcev in omogočili združevanje rezultatov vseh štirih vprašanj. Za test C1, sta sta bila povprečna ranga 36 (brez razlage) in 20 (z razlago). Povprečni rang napake brez in povprečni rang napake z razlago se torej razlikujeta, kar potrdi tudi Wilcoxonov test parov, kjer smo za alternativno hipotezo vzeli, da je povprečni rang napake z razlago nižji (p-vrednost $< 2.2 \times 10^{-16}$). Za test C2 sta bila povprečna ranga 29.5 (brez razlage) in 26.5 (z razlago), kar potrdi tudi Wilcoxonov test parov (p-vrednost $< 2.3 \times 10^{-4}$). Na podlagi teh rezultatov lahko sklepamo, da je razlaga izboljšala napovedi.

Tabela 5.3: Rezultati preverjanja kvalitete napovedi.

Poskus	**Povprečje (brez)**	**Povprečje (z razlago)**	**p.vrednost**
poskus 2, skupina A, C1	30.75	21.25	6.4×10^{-6}
poskus 2, skupina A, C2	28.25	23.75	1.5×10^{-2}
poskus 2, skupina B, C1	7.75	6.62	4.9×10^{-2}
poskus 2, skupina B, C2	8.30	5.70	1.2×10^{-2}

Tabela 5.4: Rezultati preverjanja zaupanja v lastne napovedi.

Poskus	**Povprečje (brez)**	**Povprečje (z razlago)**	**p.vrednost**
poskus 1, C1	2.58	2.89	2.5×10^{-4}
poskus 1, C2	2.73	2.78	2.4×10^{-1}
poskus 2, skupina A, C1	2.68	2.98	7.5×10^{-3}
poskus 2, skupina A, C2	2.62	2.64	4.6×10^{-1}
poskus 2, skupina B, C1	2.96	2.96	6.1×10^{-1}
poskus 2, skupina B, C2	2.71	2.87	2.1×10^{-1}

5.2.2 Poskus št. 2

Najprej navedimo rezultate poskusa, v katerem je sodelovalo 52 študentov prvega letnika (skupina A) in 14 študentov 4. letnika, za katere lahko predpostavimo, da že imajo izkušnje z odkrivanjem zakonitosti v podatkih in napovedovanjem (skupina B). Pri tem moramo omeniti, da nihče izmed teh študentov ni sodeloval pri poskusu številka 1. Poskus smo izvedli na enak način kot poskus št. 1, a z drugače sestavljenimi testi. Polovica študentov znotraj vsake skupine je prejela list C1-brez / C2, polovica pa list C2-brez / C1.

Za razliko od poskusa št. 1 je v poskusu št. 2 vsak študent reševal oba koncepta, enega je reševal brez, drugega pa z razlago. Naloga je bila torej zahtevnejša od prejšnje. Rezultate smo analizirali za vsako skupino posebej in na enak način kot pri poskusu številka 1. Ker za vzorec brez razlage nismo več imeli ustreznega vzorca z razlago, ki bi pripadal isti osebi in istemu testu, nismo uporabili preizkusa dvojic.

V tabeli 5.3 najdemo rezultate poskusa št. 2. Na podlagi rezultatov lahko sklepamo, da je razlaga izboljšala napovedi.

5.2.3 Vpliv razlage na prepričanost v lastne napovedi

Z vprašalnikom smo vzporedno preverjali tudi prepričanost v pravilnost lastnih napovedi. Uporabili smo štiristopenjsko lestvico (dodatek C), katere vrednosti smo za potrebe analize pretvorili v števila od 1 (zelo neprepričan) do 4 (zelo prepričan).

Rezultate tega poskusa, ki jih najdemo v tabeli 5.4, smo analizirali na enak način kot

range pri poskusih št. 1 in 2. Ti rezultati so podobni rezultatom za kvaliteto napovedi - prepričanost je kvečjemu višja, če je na voljo razlaga. Na podlagi tega lahko sklepamo, da uporabniki do neke mere dobro ocenjujejo točnost svojih napovedi, vendar razlike v prepričanosti niso tako prepričljive kot razlike v kvaliteti napovedi.

5.3 Pomembnost atributov Var[Λ] kot filter-metoda

V 3. poglavju smo definirali pomembnost atributov $\Lambda_i, \forall i$ za nek model f kot varianco vzorcev preko vseh možnih. Ker model f predstavlja poskus modeliranja neznanega koncepta g, nam $\Lambda_i, \forall i$ pove tudi nekaj o pomembnosti atributov za g. Pomembnost atributov lahko načeloma uporabimo tudi kot metodo za ocenjevanje in izbiro podmnožice atributov. V prvem koraku z nekim učnim algoritmom zgradimo vmesni model. V drugem koraku na modelu izvedemo algoritem 5, katerega rezultate uporabimo kot ocene atributov.

Sklepamo, da bomo za model f, katerega napovedi g so kvalitetne, dobili tudi kvalitetne ocene atributov. Za dano množico podatkov bi želeli izbrati tak učni algoritem, ki se na množici najbolje obnese. Če ni vnaprej znano, s kakšnimi množicami bomo imeli opravka, pa izberemo algoritem, ki se dobro obnese na vseh (večini) možnih konceptov. Za ilustracijo smo uporabili metaučenje bagging z odločitvenimi drevesi kot šibkimi modeli, kateremu sledi razlaga pomembnosti atributov (krajše Λ_{BagTree}).

Da bi preverili našo hipotezo, smo izvedli poskus, pri katerem smo ocene Λ_{BagTree} pomembnosti atributov primerjali z ocenami ReliefF preko vseh umetno generiranih množic. ReliefF (za regresijo RRelief) je izboljšava osnovnega algoritma za ugotavljanje pomembnosti atributov Relief, ki sta ga predlagala Kira in Rendel [46]. Izboljšave, med drugim posplošitev na večrazredne probleme, obravnavanje manjkajočih vrednosti in uporaba v regresiji, so delo Kononenka in Robnik Šikonje [50, 81]. Glavna prednost ReliefF-a je, da se dobro obnese tudi v primeru močnih interakcij med atributi. Bolj kot časi računanja nas je zanimala zmožnost pravilnega ocenjevanja pri različnih konceptih. Tako smo za ocene (R)ReliefF uporabili vse primere iz učne množice. Pri Λ_{BagTree} smo gradili po 50 dreves na množico, pomembnost atributov pa ocenili s po 5000 vzorci na atribut.

Rezultati, ki jih najdemo v tabeli 5.5, kažejo, da Λ_{BagTree} pravilno oceni pomembnost atributov pri vseh različnih konceptih, razen enega. Izjema je množica *cRedundant*, saj četrti in peti atribut, ki sta kopiji prvega in drugega, oceni za nepomembna. To je posledica načina gradnje odločitvenega drevesa, ki upošteva samo eno izmed morebitnih večih kopij atributov. Podobno se zgodi tudi pri *rRedundant*. ReliefF pravilno oceni atribute pri klasifikacijskih množicah, pri regresijskih pa RReliefF najvišjo oceno sicer podeli najpomembnejšemu atributu, manj pomembne atribute pa podcenjuje. Slednje je znana lastnost RReliefF-a [79]. RReliefF napačno označi za nepomembne tudi prve tri atribute pri regresijski različici ekskluzivnega-ali *rXor*. ReliefF in RReliefF pri redundantnih atributih obe kopiji ocenita enako.

Filter-metode običajno uporabljamo kot predprocesiranje pred uporabo učnega algo-

Tabela 5.5: Ocene atributov, ki smo jih dobili z ReliefF/RReliefF (prva vrstica), in pomembnost atributov za bagging odločitvenih/regresijskih dreves (druga vrstica) za vse umetno generirane množice podatke. V zadnjem stolpcu so podani časi računanja (v milisekundah). Vrstice, pri katerih so bili določeni atributi napačno ocenjeni, smo označili z ⇒.

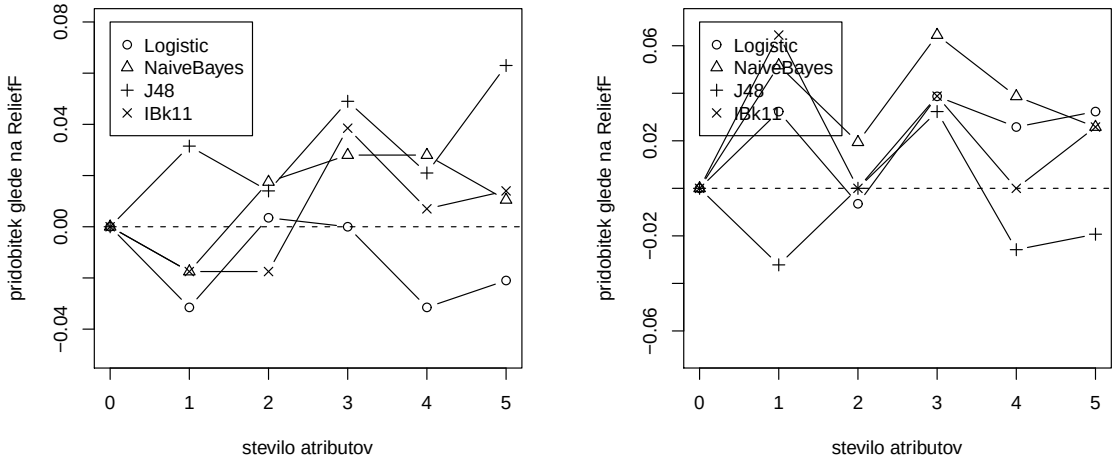
		A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	Čas
	cChess	0,0388	0,0384	-0,0120	-0,0122					1532
		0,2302	0,2390	0,0046	0,0035					1953
	cCondInd	0,0649	0,0922	0,1866	0,4605	0,0394	0,0350	0,0428	0,0475	2125
		0,0074	0,0258	0,0558	0,2157	0,0014	0,0016	0,0013	0,0012	1000
	cCross	0,0202	0,0186	-0,0012	-0,0021	-0,0014	-0,0017			2031
		0,4114	0,4104	0,0006	0,0005	0,0006	0,0008			1797
	cDisjunctN	0,1514	0,0788	0,1097	0,0120	0,0104				1766
		0,2195	0,1335	0,1650	0,0000	0,0000				453
	cGroup	0,1354	0,1333	-0,0016	-0,0009					1516
		0,3663	0,3625	0,0013	0,0020					1547
	cRandom	-0,0001	0,0011	0,0001	-0,0002					1484
		0,0120	0,0125	0,0104	0,0117					2641
	cRedundant	0,1382	0,0651	0,1343	0,1382	0,0651				1766
⇒		0,2037	0,1213	0,1672	0,0000	0,0000				453
	cSphere	0,0465	0,0507	0,0447	-0,0040	-0,0038				1765
		0,1368	0,1597	0,1418	0,0001	0,0001				1578
	cXor	0,1751	0,1735	0,1598	0,0175	0,0146	0,0132			1703
		0,3132	0,3097	0,3143	0,0006	0,0003	0,0004			875
	rDisjunctN	0,0210	0,0106	0,0164	-0,0145	-0,0210				1734
		0,2068	0,1314	0,1620	0,0000	0,0000				422
⇒	rLinear	0,0013	0,0293	0,0140	-0,0114	-0,0130				1719
		0,1358	0,6575	0,3535	0,0000	0,0000				1937
⇒	rLinNoisy	-0,0001	0,0185	0,0088	-0,0078	-0,0079				1703
		0,1357	0,6136	0,3443	0,0001	0,0001				1766
⇒	rLocLinear	0,0270	0,0036	-0,0090	-0,0115	0,0058				1703
		1,9304	0,3161	0,0795	0,0178	0,3296				1703
⇒	rNonLinPoly	0,0493	-0,0015	-0,0095	-0,0099	-0,0085				1703
		0,1351	0,0142	0,0000	0,0000	0,0000				1938
⇒	rNonLinTrig	0,0489	-0,0055	-0,0062	-0,0104	-0,0109				1719
		0,8588	0,1408	0,1398	0,0001	0,0001				2000
	rRandom	0,0001	0,0004	0,0003	0,0007					1437
		0,0037	0,0049	0,0032	0,0047					1453
⇒	rRedundant	0,0027	-0,0029	0,0137	0,0027	-0,0029				1719
⇒		0,2088	0,1260	0,1716	0,0000	0,0000				438
⇒	rXor	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000			1640
		0,3251	0,3221	0,3219	0,0014	0,0012	0,0012			985

ritma. To storimo, da zmanjšamo čase računanja ali odstranimo nepomembne atribute. V obeh primerih si želimo, da so pri vrhu tisti atributi, ki najbolj povečajo uspešnost modela. Na množicah z 10 ali več atributi smo preverili, kako $\Lambda_{\mathrm{BagTree}}$ uredi atribute in kakšen je vpliv na kvaliteto različnih modelov, če atribute dodajamo v učno množico v tem vrstnem redu. Kvaliteto modelov smo preverjali z 10-kratnim prečnim preverjanjem. Rezultate smo primerjali z algoritmom (R)ReliefF.

Na več različnih množicah se $\Lambda_{\mathrm{BagTree}}$ obnese bolje kot (R)ReliefF (glej sliki 5.2 in 5.3). Izjema sta množici annealing, kjer ReliefF daje boljše rezultate za naivnega Bayesa, in bodyfat, na kateri RReliefF daje boljše rezultate za model IBk11 (glej sliko 5.3(c)). Na preostalih množicah v tabelah 4.1 in 4.2 z 10 ali več atributi ni bilo večjih razlik med obema filter-metodama.

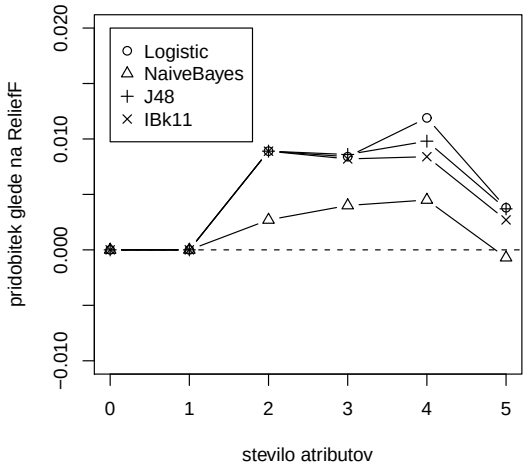
Pokazali smo, da je metoda razlage v kombinaciji z vsestranskim učnim algoritmom (v našem primeru bagging odločitvenih dreves) primerljiva z, v nekaterih pogledih celo boljša od (R)ReliefF. Naš glavni namen ni bil razviti novo filter-metodo za izbiro podmnožice atributov, temveč empirično potrditi, da dobra razlaga kvalitetnega modela posredno pove veliko tudi o samem konceptu, ki ga modeliramo.

Če bi želeli predlagano metodo za razlago uporabiti kot filter-metodo, moramo biti pozorni na sledeče. Poleg temeljitejše primerjave še z nekaterimi drugimi filter-metodami, bi bilo vredno poiskati učni algoritem, ki je bolj primeren kot Bagging. Vmesni algoritem bi lahko dinamično izbirali iz nabora algoritmov na podlagi njihove kvalitete na trenutni množici. Slednje je seveda časovno bolj zahtevno opravilo in ne moremo trditi, da je dovolj učinkovito, zato bi bile potrebne dodatne raziskave. Pri vsem tem moramo upoštevati, da metoda razlage oceni atribute glede na njihovo pomembnost za napovedi modela, za razliko od npr. naključnih gozdov, kjer je ocena atributa sestavljena iz izboljšanja kvalitete, ki jo prinese atribut. Pri kvalitetnih modelih to ni težava, pri manj kvalitetnih pa lahko privede do tega, da so nepomembni atributi ocenjeni visoko, ker so pomembni za sam model (ki se je očitno preveč prilagodil množici). Pri izbiri vmesnega učnega algoritma se izogibajmo modelom, ki so nagnjeni k prevelikem prilagajanju oz. uporabimo ustrezne tehnike, da zmanjšamo to možnost.

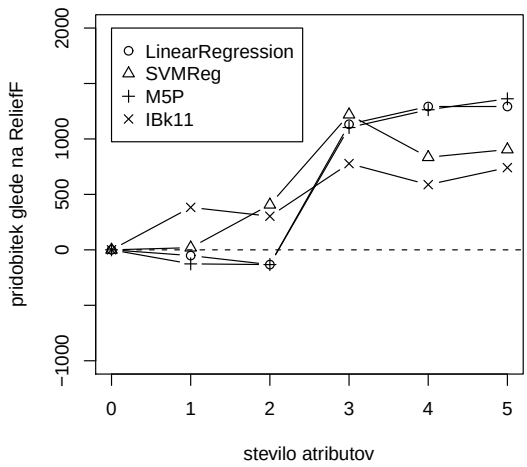


(a) breast-cancer(Lj)

(b) hepatitis

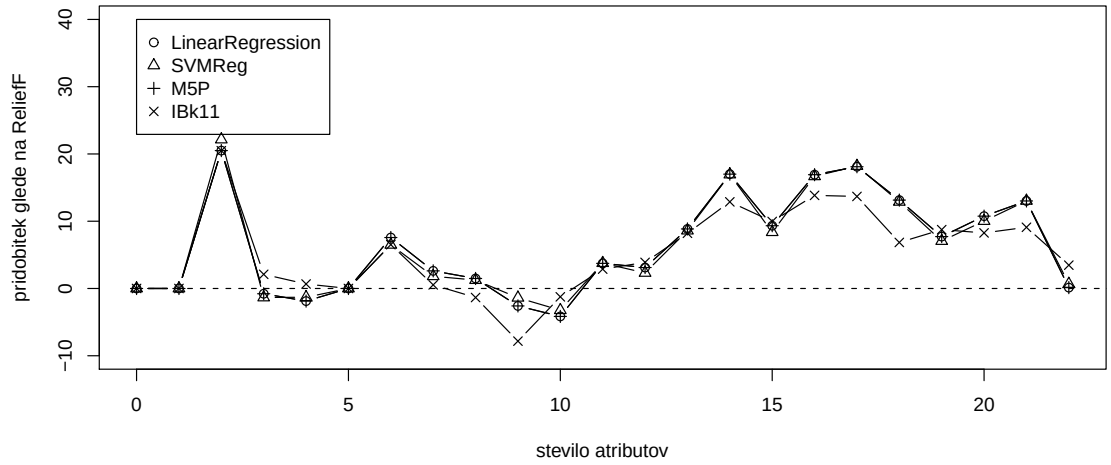


(c) mushroom

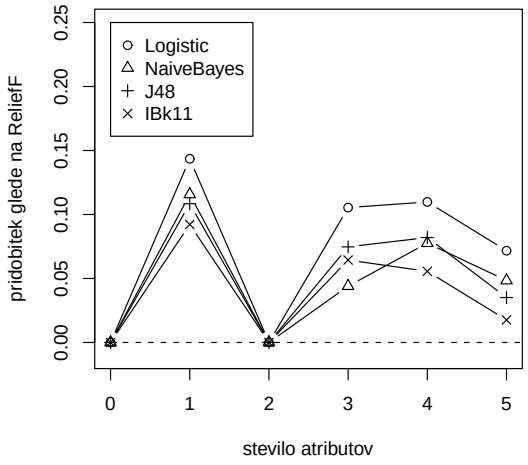


(d) pollution

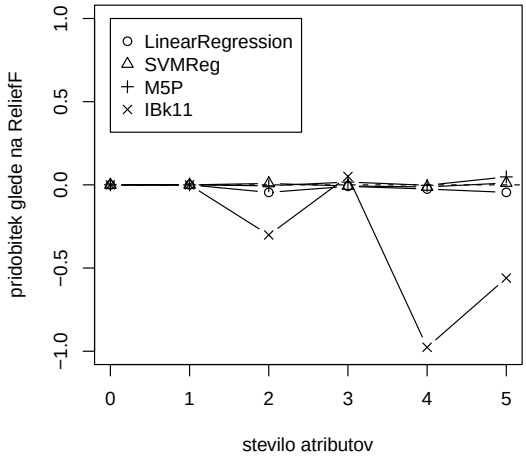
Slika 5.2: Pridobitek na točnosti (pri regresiji MSE) glede na ReliefF (pri regresiji RReliefF), če atribute dodajamo v vrstnem redu, ki ga določi $\Lambda_{\mathrm{BagTree}}$.



(a) linear50



(b) soybean



(c) bodyfat

Slika 5.3: Pridobitek na točnosti (pri regresiji MSE) glede na ReliefF (pri regresiji RReliefF), če atribute dodajamo v vrstnem redu, ki ga določi $\Lambda_{\mathrm{BagTree}}$.

Poglavje 6

Zaključek in nadaljnje delo

6.1 Zaključek

Glavni cilj tega doktorskega dela je bil **razvoj splošne metode za razlago posameznih napovedi s prispevki atributov, ki bo odpravila pomanjkljivosti obstoječih metod**. Z upoštevanjem interakcij preko vseh podmnožic atributov nam je to tudi uspelo. Želene lastnosti metode, ki so obenem tudi prednosti pred obstoječimi metodami, smo formalno utemeljili preko povezave s Shapleyevo vrednostjo. Atributu, ki ne prispeva nič, bo tako dodeljen ničeln prispevek. Atributa, ki prispevata na identičen način, bosta dobila enak prispevek. Vsota vseh prispevkov pa bo enaka razliki med napovedjo modela in pričakovano napovedjo modela, zaradi česar so prispevki implicitno normalizirani. Iz prispevkov k posamezni napovedi lahko ocenimo tudi pomembnost atributa kot celote. Pokazali smo, kako lahko ta **pristop v kombinaciji s poljubnim učnim algoritmom uporabimo kot filter-metodo** za izbiro podmnožice atributov.

Prispevke k posamezni napovedi lahko združimo v povprečne prispevke posameznih vrednosti in zgradimo **vizualizacije, podobne vizualizacijam marginalnih učinkov pri aditivnih modelih**. Če model ni aditiven, ti prispevki upoštevajo tudi interakcije. Za primer aditivnega modela pa smo dokazali, da je takšna vizualizacija enakovredna obstoječim pristopom za aditivne modele. Uporabniki aditivnih modelov torej ne izgubijo prednosti, ki jih nudijo metode za razlago, prilagojene za aditivne modele.

Osnovna različica predlagane metoda ima eksponentno časovno zahtevnost, ki je posledica pregledovanja vseh podmnožic atributov. **Za praktično rabo smo razvili aproksimacijo**, katere učinkovitost smo dodatno izboljšali z uporabo kvazi-naključnega vzorčenja in prilagajanja števila vzorcev za določen atribut varianci vzorcev tega atributa. Formalizirali smo tudi mehanizem, s katerim uravnavamo razmerje med številom vzorcev in napako aproksimacije. Z izjemo kvazi-naključnega vzorčenja, ki k učinkovitosti prispeva občutno manj od prilagajanja števila vzorcev, je implementacija aproksimacijskega algoritma kratka in preprosta, kar je z vidika praktične rabe zelo zaželeno.

Praktično preverjanje metode vključuje eksperiment, v katerem smo uporabili 45 mno-

žic podatkov 7 različnih regresijskih in 11 različnih klasifikacijskih modelov. Metoda se je izkazala kot zelo uporabno orodje za vizualizacijo modelov, odkrivanje napačno naučenih konceptov in primerjavo različnih modelov med seboj. O uporabnosti metode priča tudi **aplikacija pri problemu napovedovanja ponovitve raka dojke**, kjer so strokovnjaki onkologije potrdili uporabnost razlage in izrazili strinjanje z veliko večino prispevkov posameznih atributov. **Izvedli smo tudi študijo uporabnosti razlage**, v kateri je sodelovalo 122 študentov. Merili smo, kako razlaga vpliva na razumevanju povezav med atributi in ciljno spremenjlivko ter na točnost napovedi. Rezultati kažejo, da razlaga izboljša točnost napovedi, četudi študentje nimajo predznanja in dobijo le minimalno potrebno pojasnilo o delovanju razlage. Ne moremo pa z gotovostjo zaključiti, da razlaga poveča uporabnikovo zaupanje v pravilnost njegovih napovedi.

6.2 Razprava in nadaljnje delo

Tekom raziskav smo naleteli tudi na nekaj novih vprašanj in raziskovalnih poti, ki bi jih bilo v prihodnosti vredno raziskati. Na prvo mesto bi postavili uporabo metode kot filter-metode za izbiro podmnožice atributov. Z eksperimenti na umetnih množicah smo pokazali, da je metoda tako po uspešnosti kot po časih računanja primerljiva z metodo ReliefF, ki prav tako upošteva interakcije med atributi. Pri tem velja omeniti, da je v jedru metode ReliefF metoda najbližjih sosedov, medtem ko lahko s predlaganim pristopom uporabimo katerikoli algoritem za učenje modela. Na tem mestu omenimo tudi pomembnost atributov, ki je del algoritma za grajenje naključnih gozdov in se pogosto uporablja za izbiro podmnožice atributov pri množicah z velikim številom atributov. Ta metoda, za razliko od predlagane metode, pomembnost atributa določi na podlagi njegovega vpliva na točnosti modela in ne na podlagi vpliva na napoved modela. Uporaba predlagane metode v kombinaciji z naključnimi gozdovi (brez internega računanja pomembnosti) je časovno enako zahtevna kot gradnja naključnih gozdov z internim računanjem pomembnosti. Zanimivo bi bilo torej preveriti, katera metoda bolje uredi atribute po pomembnosti. Še posebej, če po izbiri podmnožice atributov za učenje končnega modela uporabimo nek drug algoritem namesto naključnih gozdov.

V tem doktorskem delu smo se ukvarjali s prispevki posameznih atributov. Druga razširjena oblika razlage pa so odločitvena drevesa (pravila, tabele,...), ki so v določenih primerih bolj, v drugih pa manj primerna od razlage s prispevki. Kot pomembno odprto vprašanje in iziv bi torej izpostavili problem, kako združiti obe obliki razlage. Kot izhodiščno idejo bi izpostavili problem učinkovitega opisa nekega neznanega modela s čim bolj preprostim odločitvenim drevesom, ki ima v listih posplošene aditivne modele. Z drugimi besedami, kako model čim bolje razložiti s čim manjšo množico razlag, kot smo jih ponudili v tem delu, in pravili, na podlagi katerih bo uporabnik za posamezen primer izbral pravilno razlago.

Omeniti velja tudi primernost metode za razlago modelov pri delu s podatkovnimi to-

kovi. Dve posebnosti dela z modeli podatkovnih tokov sta potencialno stalno spreminjanje modela ter potreba po učinkovitem in sprotnem računanju. Ker je računanje prispevkov inkrementalno, lahko v vsaki časovni rezini izračunamo le toliko vzorcev, kolikor nam jih čas dopušča. Starejše vzorce lahko po potrebi zavržemo, uporabimo drseče okno, obtežimo obratno sorazmerno s časom ali uporabimo katero izmed preostalih tehnik, ki se pri delu s podatkovnimi tokovi uporabijo za oblikovanje trenutne učne množice za gradnjo modela. Tako dosežemo, da se bo razlaga spreminjala skupaj s spreminjanjem modela.

Rezultati študije razumljivosti razlage so vzpodbudni. Kljub minimalnim pojasnilom in predznanju uporabnikov razlaga v povprečju poveča točnost napovedi in ne zmanjša (kvečjemu poveča) uporabnikovo zaupanje v lastne napovedi. Seveda pa študija ni izčrpna in pušča veliko prostora za izboljšave. V prihodnosti bi bilo smiselno vključiti dodatne koncepte, predvsem bolj zapletene koncepte. Po zgledu podobnih študij bi lahko vključili interakcijo uporabnika z modelom in razlago, napovedovanje ne samo ciljne spremenljivke, temveč tudi manjkajočih vrednosti atributov, ter spremljanje časov reševanja. Pri tem pa se moramo zavedati, da bolj zapleteni poskusi zahtevajo tudi večjo množico uporabnikov, kar v praksi predstavlja precejšen problem. Vsekakor pa velja še enkrat poudariti, da na področju odkrivanja zakonitosti v podatkih primanjkuje takšnih raziskav razumljivosti različnih oblik razlage oz. reprezentacij znanja.

Literatura

[1] C. H. Achen. *Intepreting and Using Regression.* Sage Publications, 1982.

[2] H. Allahyari and N. Lavesson. User-oriented assessment of classification model understandability. In *Proceedings of the 11th Scandinavian Conference on Artificial Intelligence - SCAI 2011*, pages 11–19, 227.

[3] I. Alvarez. Explaining the result of a decision tree to the end-user. In *Proceedings of the 16th European Conference on Artificial Intelligence*, pages 411–415, 2004.

[4] R. Andrews, J. Diederich, and A. B. Tickle. Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-Based Systems*, 8:373–389, 1995.

[5] A. Asuncion and D. J. Newman. UCI Machine Learning Repository, `http://www.ics.uci.edu/~mlearn/MLRepository.html`, 2009.

[6] N. H. Barakat and A. P. Bradley. Rule extraction from support vector machines: A sequential covering approach. *IEEE Trans. on Knowl. and Data Eng.*, 19(6):729–741, 2007.

[7] B. Becker, R. Kohavi, and D. Sommerfield. Visualizing the simple Bayesian classier. In *KDD Workshop on Issues in the Integration of Data Mining and Data Visualization*, 1997.

[8] J. Blanchard, F. Guillet, and H. Briand. Exploratory visualization for association rule rummaging. In *KDD-03 Workshop on Multimedia Data Mining (MDM-03*, pages 107–114, 2003.

[9] J. Blanchard, F. Guillet, and H. Briand. A user-driven and quality-oriented visualization for mining association rules. In *ICDM '03: Proceedings of the Third IEEE International Conference on Data Mining*, page 493, Washington, DC, USA, 2003. IEEE Computer Society.

[10] J. Blanchard, F. Guillet, and H. Briand. Interactive visual exploration of association rules with rule-focusing methodology. *Knowledge and Information Systems*, 13:43–75, 2007.

[11] M. Bohanec and I. Bratko. Trading accuracy for simplicity in decision trees. *Mach. Learn.*, 15(3):223–250, 1994.

[12] C. C. Bojarczuk, A. S. D. Setembro, H. S. Lopes, A. A. Freitas, and R. I. Conceio. Discovering comprehensible classification rules using genetic programming: a case study in a medical domain, 1999.

[13] L. Breiman. Random forests. *Machine Learning Journal*, 45:5–32, 2001.

[14] H. Brighron and C. Mellish. Advances in instance selection for instance-based learning algorithms. *Data Mining and Knowledge Discovery*, 6(2):153–172, 2002.

[15] J. R. Cano, F. Herrera, and M. Lozano. Evolutionary stratified training set selection for extracting classification rules with trade off precision-interpretability. *Data & Knowledge Engineering*, 60(1):90–108, 2007.

[16] J. Castro, D. Gómez, and J. Tejada. Polynomial calculation of the shapley value based on sampling. *Computers and Operations Research*, 2008. (in print, doi: 10.1016/j.cor.2008.04.004).

[17] A. Chevan and M. Sutherland. Hierarchical partitioning. *The American Statistician*, 45(2):90–96, 1991.

[18] S. Cohen, G. Dror, and E. Ruppin. Feature selection via coalitional game theory. *Neural Computation*, 19(7):1939–1961, 2007.

[19] P. Cortez, J. Teixeira, A. Cerdeira, F. Almeida, T. Matos, and J. Reis. Using data mining for wine quality assessment. In *Discovery Science*, pages 66–79, 2009.

[20] M. W. Craven and J. Shavlik. Using sampling and queries to extract rules from trained neural networks. In *ICDM*, pages 37–45, 1994.

[21] V. De Angelis, G. Felici, and G. Mancinelli. *Data Mining and Knowledge Discovery Approaches Based on Rule Induction Techniques*, chapter Feature Selection for Data Mining, pages 227–252. Massive Computing. Springer US, 2006.

[22] I. De Falco, A. Della Cioppa, A. Iazzetta, and E. Tarantino. An evolutionary approach for automatically extracting intelligible classification rules. *Knowledge and Information Systems*, 7:179–201, 2005.

[23] A. L. de Santana, C. Frances, C. A. Rocha, S. V. Carvalho, N. L. Vijaykumar, L. P. Rego, and J. C. Costa. Strategies for improving the modeling and interpretability of bayesian networks. *Data & Knowledge Engineering*, 63(1):91–107, 2007.

[24] S. Dehuri and R. Mall. Predictive and comprehensible rule discovery using a multi-objective genetic algorithm. *Knowledge Based Systems*.

[25] S. Dehuri, S. Patnaik, A. Ghosh, and R. Mall. Application of elitist multi-objective genetic algorithm for classification rule generation. *Applied Soft Computing*, 8(1):477–487, 2008.

[26] J. Demšar, B. Zupan, G. Leban, and T. Curk. Orange: From experimental machine learning to interactive data mining. In *PKDD'04*, pages 537–539, 2004.

[27] S. Ding, W. Jia, C. Su, F. Jin, and Z. Shi. A survey on statistical pattern feature extraction. In *Proceedings of the 4th international conference on Intelligent Computing: Advanced Intelligent Computing Theories and Applications - with Aspects of Artificial Intelligence*, LNCS, pages 701–708, 2008.

[28] R. Doerfler. The lost art of nomography. *The UMAP Journal*, 30.4:457–494, 2009.

[29] U. Fayyad, G. Piatetsky-shapiro, and P. Smyth. From data mining to knowledge discovery in databases. *AI Magazine*, 17:37–54, 1996.

[30] B. Feldman. Relative importance and value. Tehnično poročilo. Dostopno na `http://www.prismanalytics.com/docs/RelativeImportance050319.pdf`, 2005.

[31] M. Grochowski and N. Jankowski. *Artificial Intelligence and Soft Computing - ICAISC 2004*, chapter Comparison of Instance Selection Algorithms II. Results and Comments, pages 580–585. LNCS. Springer Berlin, 2004.

[32] U. Grömping. Estimators of relative importance in linear regression based on variance decomposition. *The American Statistician*, 61:139–147, 2007.

[33] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[34] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh. *Feature Extraction: Foundations and Applications*. Springer, 2006.

[35] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, 2009.

[36] L. Hamel. Visualization of support vector machines with unsupervised learning. In *Computational Intelligence in Bioinformatics and Computational Biology*, pages 1–8. IEEE, 2006.

[37] F. E. Harrell. *Regression Modeling Strategies, with Applications to Linear Models, Survival Analysis and Logistic Regression*. Springer, 2001. ISBN 0-387-95232-2.

[38] T. Hastie and R. Tibshirani. Generalized additive models. *Statistical Science*, 1:297–310, 1986.

[39] M. Holena. Extraction of logical rules from data by means of piecewise-linear neural networks. In *LNCS: Discovery Science*. Springer, 2007.

[40] J. Huysmans, K. Dejaeger, C. Mues, J. Vanthienen, and B. Baesens. An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. *Decision Support Systems*, 51(1):141 – 154, 2011.

[41] P. Jaeckel. *Monte Carlo Methods in Finance*. Wiley, New York, 2002.

[42] A. Jakulin, M. Možina, J. Demšar, I. Bratko, and B. Zupan. Nomograms for visualizing support vector machines. In *KDD '05: 11th ACM SIGKDD*, pages 108–117. ACM, 2005.

[43] N. Jankowski and M. Grochowski. *Artificial Intelligence and Soft Computing - ICAISC 2004*, chapter Comparison of Instances Seletion Algorithms I. Algorithms Survey, pages 598–603. LNCS. Springer Berlin, 2004.

[44] M. W. Kattan, J. A. Eastham, A. M. Stapleton, T. M. Wheeler, and P. T. Scardino. A preoperative nomogram for disease recurrence following radical prostatectomy for prostate cancer. *Journal of the National Cancer Institute*, 90:766–771, 1998.

[45] A. Keinan, B. Sandbank, C. C. Hilgetag, I. Meilijson, and E. Ruppin. Fair attribution of functional contribution in artificial and biological networks. *Neural Computation*, 16(9):1887–1915, 2004.

[46] K. Kira and L. A. Rendell. A practical approach to feature selection. In D. H. Sleeman and P. Edwards, editors, *Ninth International Workshop on Machine Learning*, pages 249–256. Morgan Kaufmann, 1992.

[47] D. E. Knuth. *The Art of Computer Programming, volume 2: Seminumerical Algorithms*. Addison-Wesley, 1998.

[48] R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence journal*, 97(1–2):273–324, 1997.

[49] I. Kononenko. Inductive and bayesian learning in medical diagnosis. *Applied Artificial Intelligence*, 7:317–337, 1993.

[50] I. Kononenko. Estimating attributes: Analysis and extensions of Relief. In F. Bergadano and L. D. Raedt, editors, *European Conference on Machine Learning*, pages 171–182. Springer, 1994.

[51] I. Kononenko. Machine learning for medical diagnosis: History, state of the art and perspective. *Artificial Intelligence in Medicine*, 23:89–109, 2001.

[52] R. Krishnan, G. Sivakumar, and P. Bhattacharya. Extracting decision trees from trained neural networks. *Pattern Recognition*, 32:1999–2009, 1999.

[53] W. Kruskal. Correction to relative importance by averaging over orderings. *The American Statistician*, 41:341, 1987.

[54] W. Kruskal. Relative importance by averaging over orderings. *The American Statistician*, 41:6–10, 1987.

[55] M. Kukar and C. Grošelj. Supporting diagnostics of coronary artery disease with neural networks. In A. Dobnikar, U. Lotric, and B. Šter, editors, *Adaptive and Natural Computing Algorithms*, volume 6593 of *Lecture Notes in Computer Science*, pages 80–89. Springer Berlin / Heidelberg, 2011.

[56] V. Lemaire, R. Féraud, and N. Voisine. Contact personalization using a score understanding method. In *International Joint Conference on Neural Networks (IJCNN)*, 2008.

[57] B. Y. Lim, A. K. Dey, and D. Avrahami. Why and why not explanations improve the intelligibility of context-aware intelligent systems. In *Proceedings of the 27th international conference on Human factors in computing systems*, CHI '09, pages 2119–2128, New York, NY, USA, 2009. ACM.

[58] S. Lipovetsky and M. Conklin. Analysis of regression in game theory approach. *Applied Stochastic Models in Business and Industry*, 17:319–330, 2001.

[59] H. Liu and H. Motoda, editors. *Feature Extraction, Construction and Selection*, The Springer International Series in Engineering and Computer Science. Springer, 1998.

[60] H. Liu and H. Motoda. *Feature Selection for Knowledge Discovery and Data Mining*. Kluwer Academic Publishers, Norwell, MA, USA, 1998.

[61] H. Liu and H. Motoda, editors. *Instance Selection and Construction for Data Mining*, The Springer International Series in Engineering and Computer Science. Springer, 2001.

[62] H. Liu and H. Motoda. On issues of instance selection. *Data Mining and Knowledge Discovery*, 6(2):114–130, 2002.

[63] J. Lubsen, J. Pool, and E. van der Does. A practical device for the application of a diagnostic or prognostic function. *Methods of Information in Medicine*, 17:127–129, 1978.

[64] D. Madigan, K. Mosurski, and R. G. Almond. Graphical explanation in belief networks. *In Journal of Computational and Graphical Statistics*, 6:160–181, 1997.

[65] D. Martens, B. Baesens, T. Van Gestel, and J. Vanthienen. Comprehensible credit scoring models using rule extraction from support vector machines. *European Journal of Operational Research*, 183(3):1466–1476, 2007.

[66] G. Melli. The datgen dataset generator. `http://www.datasetgenerator.com`.

[67] S. Moretti and F. Patrone. Transversality of the Shapley value. *TOP*, 16(1):1–41, 2008.

[68] M. Možina, J. Demšar, M. Kattan, and B. Zupan. Nomograms for visualization of naive Bayesian classifier. In *PKDD 2004*, pages 337–348. Springer-Verlag, 2004.

[69] R. Nayak. Generating rules with predicates, terms and variables from the pruned neural networks. *Neural Networks*, 22(4):405–414, 2009.

[70] H. Niederreiter. Low-discrepancy and low-dispersion sequences. *Journal of Number Theory*, 30(1):51 – 70, 1988.

[71] H. Niederreiter. *Random number generation and quasi-Monte Carlo methods*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1992.

[72] F. Poulet. Svm and graphical algorithms: A cooperative approach. In *4th IEEE ICDM*, pages 499–502, 2004.

[73] M. Pregeljc. *The application of data mining to explore the connections between the quality of organization of companies and their business results (Uporaba odkrivanja zakonistosti v podatkih za iskanje povezav med kakovostjo organizacije združb in njihovimi poslovnimi izidi)*. PhD thesis, University of Ljubljana, Faculty of Computer and Information Science, Ljubljana, 2011. (available at http://eprints.fri.uni-lj.si/1358/1/Pregeljc1.pdf).

[74] M. D. Radović, N. D. Filipović, Z. Bosnić, P. Vračar, and I. Kononenko. Mining data from hemodynamic simulations for generating prediction and explanation models. In *The 10th IEEE International Conference on Information Technology and Applications in Biomedicine: Emerging technologies for patient specific healthcare*, pages 1–4, 2010.

[75] C. P. Rainsford and J. F. Roddick. Visualisation of temporal interval association rules. In *IDEAL '00: Proceedings of the Second International Conference on Intelligent Data Engineering and Automated Learning, Data Mining, Financial Engineering, and Intelligent Agents*, pages 91–96, London, UK, 2000. Springer-Verlag.

[76] D. Randall Wilson and T. R. Martinez. Reduction techniques for instance-based learning algorithms. *Machine Learning*, 38(3):257–286, 2000.

[77] T. Reinartz. A unifying view on instance selection. *Data Mining and Knowledge Discovery*, 6(2):191–210, 2002.

[78] G. Ridgeway, D. Madigan, and T. Richardson. Interpretable boosted nai"ve bayes classification. In *4 th Int Conf on Knowledge Discovery and Data Mining - KDD98, pp 101-104*, 1998.

[79] M. Robnik Šikonja and I. Kononenko. Comprehensible interpretation of Relief's estimates. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pages 433–440, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[80] M. Robnik-Šikonja, A. Likas, C. Constantinopoulos, I. Kononenko, and E. Štrumbelj. Efficiently explaining decisions of probabilistic rbf classification networks. In A. Dobnikar, U. Lotric, and B. Šter, editors, *Adaptive and Natural Computing Algorithms*, volume 6593 of *Lecture Notes in Computer Science*, pages 169–179. Springer Berlin / Heidelberg, 2011.

[81] M. Robnik-Šikonja and I. Kononenko. An adaptation of Relief for attribute estimation in regression. In D. H. Fisher, editor, *Fourteenth International Conference on Machine Learning*, pages 296–304. Morgan Kaufmann, 1997.

[82] M. Robnik-Šikonja and I. Kononenko. Explaining classifications for individual instances. *IEEE TKDE*, 20:589–600, 2008.

[83] L. S. Shapley. *A Value for n-person Games*, volume II of *Contributions to the Theory of Games*. Princeton University Press, 1953.

[84] Q. Shen and A. Chouchoulas. A rough-fuzzy approach for generating classification rules, 2002.

[85] C. Strobl, A.-L. Boulesteix, T. Kneib, T. Augustin, and A. Zeileis. Conditional variable importance for random forests. *BMC Bioinformatics*, 9(307), 2008.

[86] C. Strobl, A.-L. Boulesteix, A. Zeileis, and T. Hothorn. Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC Bioinformatics*, 8(25), 2007.

[87] J. Stufken. On hierarchical partitioning. *The American Statistician*, 46:70–71, 1992.

[88] D. Szafron, B. Poulin, R. Eisner, P. Lu, R. Greiner, D. Wishart, A. Fyshe, B. Pearcy, C. Macdonell, and J. Anvik. Visual explanation of evidence in additive classifiers. In *Proceedings of Innovative Applications of Artificial Intelligence*, 2006.

[89] S. Thrun, J. Bala, E. Bloedorn, I. Bratko, B. Cestnik, J. Cheng, K. D. Jong, S. Dzeroski, R. Hamann, K. Kaufman, S. Keller, I. Kononenko, J. Kreuziger, R. Michalski, T. Mitchell, P. Pachowicz, B. Roger, H. Vafaie, W. V. de Velde, W. Wenzel, J. Wnek, and J. Zhang. The MONK's problems: A performance comparison of different learning algorithms. Technical Report CMU-CS-91-197, Carnegie Mellon University, Computer Science Department, Pittsburgh, PA, 1991.

[90] G. Towell and J. W. Shavlik. Extracting refined rules from knowledge-based neural networks, machine learning. *Machine Learning*, 13:71–101, 1993.

[91] B. Ustün, W. Melssen, and L. Buydens. Visualisation and interpretation of support vector regression models. *Anal Chim Acta*, 595(1-2):299–309, 2007.

[92] M. J. van der Laan. Statistical inference for variable importance. *The International Journal of Biostatistics*, 2(1), 2006.

[93] N. A. Vien, N. H. Viet, T. Chung, H. Yu, S. Kim, and B. H. Cho. Vrifa: A nonlinear SVM visualization tool using nomogram and localized radial basis function (LRBF) kernels. In *CIKM*, pages 2081–2082, 2009.

[94] E. Štrumbelj, Z. Bosnić, B. Zakotnik, C. Grašič-Kuhar, and I. Kononenko. Explanation and reliability of breast cancer recurrence predictions. *Knowledge and Information Systems*, 24(2):305–324, 2010.

[95] E. Štrumbelj and I. Kononenko. Towards a model independent method for explaining classification for individual instances. In *Proceedings of Data Warehousing and Knowledge Discovery*, LNCS, pages 273–282. Springer Berlin, 2008.

[96] E. Štrumbelj and I. Kononenko. An efficient explanation of individual classifications using game theory. *Journal of Machine Learning Research*, 11:1–18, 2010.

[97] E. Štrumbelj and I. Kononenko. A general method for visualizing and explaining black-box regression models. In A. Dobnikar, U. Lotric, and B. Ster, editors, *ICANNGA (2)*, volume 6594 of *Lecture Notes in Computer Science*, pages 21–30. Springer, 2011.

[98] E. Štrumbelj, I. Kononenko, and M. Robnik Šikonja. Explaining instance classifications with interactions of subsets of feature values. *Data & Knowledge Engineering*, 68(10):886–904, 2009.

[99] B. P. Welford. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 4(3):419–420, 1962.

[100] P. C. Wong, P. Whitney, and J. Thomas. Visualizing association rules for text mining. In *INFOVIS '99: Proceedings of the 1999 IEEE Symposium on Information Visualization*, page 120, Washington, DC, USA, 1999. IEEE Computer Society.

[101] D. S. Yeung and H. S. Fong. A knowledge matrix representation for a rule-mapped neural network. *Neurocomputing*, 7:123–144, 1995.

[102] Y. Zhang, H. Su, T. Jia, and J. Chu. Rule extraction from trained support vector machines. In *PAKDD*, pages 61–70, 2005.

[103] A. Zien, N. Krämer, S. Sonnenburg, and G. Rätsch. The feature importance ranking measure. In *ECML PKDD 2009, Part II*, pages 694–709. Springer-Verlag, 2009.

Dodatek A

Kvaliteta modelov, variance in časi

Tabela A.1: Točnosti modelov.

	Logistic	J48	MPerceptron	SMO	NaiveBayes	RandomForest	AdaBoostM1J48	AdaBoostM1NB	Bagging	IBk1	IBk11
cChess	0.495	0.516	0.756	0.487	0.487	0.974	0.516	0.494	0.984	0.769	0.773
cCondInd	0.910	0.898	0.891	0.897	0.910	0.892	0.897	0.910	0.904	0.895	0.905
cCross	0.499	0.532	0.967	0.510	0.532	0.999	0.532	0.535	1.000	0.550	0.548
cDisjunctB	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
cDisjunctN	0.903	1.000	0.993	0.900	0.936	1.000	1.000	0.956	1.000	0.940	0.958
cGroup	0.318	0.342	0.981	0.284	0.314	0.996	0.342	0.314	0.997	0.992	0.995
cRandom	0.496	0.475	0.482	0.507	0.494	0.494	0.475	0.494	0.503	0.489	0.511
cRedundant	0.902	1.000	0.997	0.896	0.938	1.000	1.000	0.960	1.000	0.977	0.980
cSphere	0.521	0.917	0.898	0.521	0.919	0.940	0.936	0.959	0.938	0.822	0.859
cXor	0.496	0.902	0.902	0.460	0.497	0.902	0.902	0.497	0.902	0.902	0.902
anneal	0.993	0.988	0.993	0.970	0.862	0.991	0.993	0.938	0.986	0.984	0.957
breast-cancer(Lj)	0.682	0.738	0.689	0.696	0.738	0.671	0.664	0.689	0.713	0.731	0.727
hepatitis	0.826	0.781	0.800	0.852	0.832	0.832	0.832	0.858	0.826	0.819	0.819
iris	0.973	0.947	0.967	0.960	0.967	0.947	0.933	0.933	0.947	0.953	0.967
monks1	0.750	0.981	1.000	0.750	0.750	0.984	1.000	0.736	0.986	0.993	0.916
monks2	0.653	0.671	1.000	0.671	0.664	0.438	0.613	0.664	0.664	0.592	0.644
monks3	1.000	1.000	1.000	1.000	0.972	1.000	1.000	0.995	1.000	0.988	0.981
mushroom	1.000	1.000	1.000	1.000	0.957	1.000	1.000	1.000	1.000	1.000	0.999
nursery	0.925	0.970	0.996	0.932	0.903	0.982	0.995	0.919	0.975	0.983	0.983
soybean	0.931	0.908	0.906	0.922	0.924	0.892	0.933	0.919	0.868	0.908	0.856
zoo	0.950	0.921	0.960	0.921	0.950	0.970	0.960	0.960	0.931	0.960	0.881

Tabela A.2: Povprečni čas računanja enega vzorca (v milisekundah).

	Logistic	J48	MPerceptron	SMO	NaiveBayes	RandomForest	AdaBoostM1J48	AdaBoostM1NB	Bagging	IBk1	IBk11
cChess	0.0054	0.0010	0.0090	0.0026	0.0090	0.0096	0.0011	0.0247	0.0170	0.5049	0.6707
cCondInd	0.0059	0.0017	0.0118	0.0046	0.0040	0.0055	0.0092	0.0219	0.0056	0.5886	0.9539
cCross	0.0051	0.0012	0.0105	0.0030	0.0127	0.0083	0.0012	0.0466	0.0120	0.6979	1.0186
cDisjunctB	0.0053	0.0013	0.0096	0.0039	0.0064	0.0025	0.0015	0.0071	0.0036	0.5352	0.5758
cDisjunctN	0.0057	0.0014	0.0101	0.0030	0.0119	0.0026	0.0014	0.1074	0.0037	0.5507	0.7749
cGroup	0.0055	0.0011	0.0099	0.0030	0.0136	0.0088	0.0012	0.0137	0.0125	0.4669	0.6153
cRandom	0.0047	0.0011	0.0089	0.0025	0.0092	0.0149	0.0011	0.0177	0.0226	0.4542	0.6361
cRedundant	0.0052	0.0014	0.0099	0.0028	0.0126	0.0026	0.0014	0.1147	0.0037	0.6940	0.8128
cSphere	0.0049	0.0021	0.0098	0.0028	0.0114	0.0067	0.0160	0.0999	0.0107	0.5616	0.7752
cXor	0.0053	0.0015	0.0105	0.0042	0.0032	0.0046	0.0068	0.0034	0.0075	0.5413	0.6767
anneal	0.0319	0.0092	0.1458	0.0472	0.0825	0.0113	0.0363	0.7490	0.0112	0.9531	1.4984
breast-cancer(Lj)	0.0092	0.0016	0.0537	0.0092	0.0043	0.0041	0.0069	0.0340	0.0044	0.1334	0.2283
hepatitis	0.0091	0.0032	0.0211	0.0079	0.0168	0.0082	0.0140	0.1391	0.0070	0.1411	0.2365
iris	0.0060	0.0014	0.0098	0.0030	0.0166	0.0025	0.0091	0.1566	0.0034	0.0629	0.0845
monks1	0.0061	0.0013	0.0164	0.0050	0.0031	0.0035	0.0041	0.0240	0.0048	0.1059	0.1986
monks2	0.0061	0.0011	0.0164	0.0050	0.0031	0.0042	0.0068	0.0105	0.0047	0.1057	0.2025
monks3	0.0062	0.0014	0.0166	0.0051	0.0032	0.0031	0.0014	0.0238	0.0038	0.1056	0.1944
mushroom	0.0194	0.0041	0.2135	0.0207	0.0110	0.0059	0.0042	0.0811	0.0063	7.4118	8.9389
nursery	0.0135	0.0025	0.0291	0.0132	0.0083	0.0054	0.0187	0.0577	0.0070	2.7610	4.1932
soybean	0.1232	0.0153	0.1403	0.4327	0.0964	0.0177	0.1042	0.9250	0.0172	1.0799	1.4331
zoo	0.0196	0.0037	0.0221	0.0174	0.0284	0.0049	0.0129	0.1879	0.0067	0.0864	0.1402

Tabela A.3: Povprečne variance $\overline{\sigma^2}$.

	Logistic	J48	MPerceptron	SMO	NaiveBayes	RandomForest	AdaBoostM1J48	AdaBoostM1NB	Bagging	IBk1	IBk11
cChess	0.0001	0.0000	0.1030	0.1464	0.0001	0.1509	0.0000	0.0001	0.0973	0.3200	0.0600
cCondInd	0.0196	0.0305	0.0429	0.0312	0.0193	0.0534	0.0466	0.0452	0.0226	0.0541	0.0166
cCross	0.0001	0.0000	0.1233	0.1256	0.0005	0.0949	0.0000	0.0012	0.1170	0.3238	0.0282
cDisjunctB	0.0564	0.0564	0.0557	0.0564	0.0245	0.0563	0.0564	0.0245	0.0564	0.1117	0.0578
cDisjunctN	0.0351	0.0800	0.0788	0.0926	0.0289	0.0797	0.0799	0.0699	0.0797	0.1152	0.0544
cGroup	0.0000	0.0000	0.1866	0.0396	0.0000	0.1657	0.0000	0.0000	0.1616	0.1883	0.1805
cRandom	0.0001	0.0000	0.0064	0.1615	0.0002	0.0554	0.0000	0.0001	0.0120	0.4056	0.0323
cRedundant	0.0256	0.0801	0.0962	0.0936	0.0384	0.0440	0.0801	0.0702	0.0796	0.1166	0.0756
cSphere	0.0003	0.0970	0.0787	0.1120	0.0060	0.0726	0.1107	0.1002	0.0590	0.2069	0.0379
cXor	0.0001	0.1528	0.1605	0.1205	0.0001	0.1578	0.1540	0.0001	0.1491	0.1555	0.1561
anneal	0.0153	0.0037	0.0035	0.0002	0.0046	0.0013	0.0045	0.0087	0.0031	0.0103	0.0038
breast-cancer(Lj)	0.0049	0.0045	0.0711	0.0518	0.0062	0.0234	0.0584	0.0198	0.0036	0.0628	0.0052
hepatitis	0.0046	0.0098	0.0170	0.0104	0.0062	0.0026	0.0167	0.0103	0.0007	0.0282	0.1335
iris	0.2126	0.0610	0.1370	0.0245	0.1919	0.0453	0.0941	0.1747	0.0406	0.2126	0.1335
monks1	0.0139	0.0139	0.1311	0.0313	0.0134	0.1189	0.0125	0.0125	0.0972	0.1344	0.0271
monks2	0.0009	0.0000	0.2431	0.0000	0.0008	0.1174	0.0803	0.0102	0.0034	0.2441	0.0339
monks3	0.0639	0.0639	0.0632	0.0638	0.0345	0.0616	0.0639	0.0639	0.0639	0.0621	0.0213
mushroom	0.0247	0.0136	0.0238	0.0393	0.0210	0.0056	0.0136	0.0330	0.0122	0.0276	0.0172
nursery	0.0278	0.0278	0.0276	0.0042	0.0277	0.0278	0.0278	0.0272	0.0278	0.0272	0.0197
soybean	0.0056	0.0024	0.0085	0.0000	0.0026	0.0010	0.0034	0.0051	0.0025	0.0043	0.0013
zoo	0.0140	0.0099	0.0076	0.0006	0.0066	0.0031	0.0104	0.0148	0.0057	0.0222	0.0099

Tabela A.4: MSE modelov (relativno glede na najslabši MSE).

	LinearRegression	M5P	MPerceptron	SVMreg	Bagging	IBk1	IBk11	najvišji MSE
rDisjunctB	0.4782	0.0006	0.0000	1.0000	0.0000	0.0000	0.0000	0.1313
rDisjunctN	0.4905	0.0038	0.2620	1.0000	0.0015	0.1655	0.2880	0.2066
rLinear	0.0000	0.0000	0.0004	0.0001	0.2099	0.4482	1.0000	0.0376
rLinNoisy	0.2755	0.2755	0.3119	0.2763	0.3786	0.4861	1.0000	0.0839
rLocLinear	0.9956	0.0003	0.0792	1.0000	0.0051	0.3609	0.4875	0.3691
rNonLinPoly	0.7982	0.0048	0.0002	1.0000	0.0038	0.2858	0.3918	0.0696
rNonLinTrig	0.9579	0.0437	0.0005	1.0000	0.0326	0.3584	0.5344	0.4554
rRandom	0.4906	0.4906	0.5389	0.6114	0.5033	0.5256	1.0000	0.5115
rRedundant	0.4885	0.0038	0.2048	1.0000	0.0015	0.0810	0.1112	0.2067
rXor	0.5327	0.2182	0.2220	1.0000	0.1942	0.1969	0.1969	0.4703
autoMpg	0.6518	0.5107	0.9625	0.7100	0.7565	0.8909	1.0000	13.6912
bodyfat	0.1229	0.1103	0.1391	0.1107	0.1551	0.7579	1.0000	15.4167
concrete	0.9076	0.3192	0.4876	1.0000	0.2724	0.7559	0.6731	120.2621
elevators	0.4211	0.2632	0.2632	0.4737	0.4211	0.6842	1.0000	1.9×10^{-5}
fishcatch	0.3860	0.1527	0.1307	0.3628	0.3637	1.0000	0.6321	24970.3000
fruitfly	0.4312	0.4312	0.7092	0.4567	0.4528	0.4909	1.0000	590.3522
housing	0.8172	0.5382	0.5949	0.8952	0.4632	1.0000	0.6833	29.02516
machinecpu	0.5743	0.3679	0.2912	0.7242	0.7851	1.0000	0.4558	8457.0200
pollution	0.2479	0.1877	1.0000	0.2385	0.3161	0.3232	0.3990	7637.9670
stock	0.9529	0.1474	0.2004	1.0000	0.1530	0.1346	0.0888	5.7433
wine	0.9014	0.8281	1.0000	0.9099	0.7008	0.8230	0.9280	0.6294
wisconsin	0.3023	0.2978	1.0000	0.2984	0.2677	0.2617	0.4860	3819.0680

Tabela A.5: Povprečni čas računanja enega vzorca (v milisekundah).

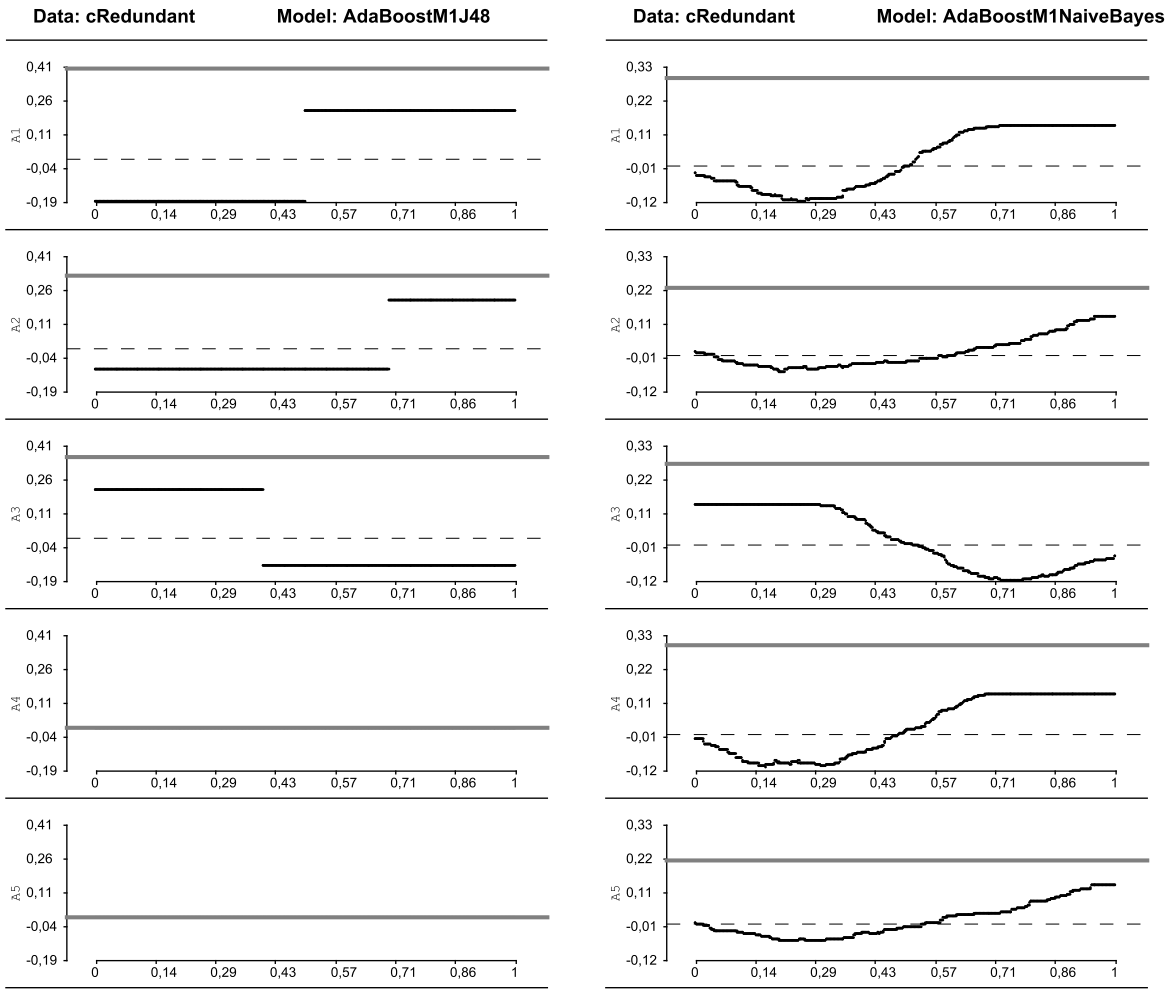
	LinearRegression	M5P	MPerceptron	SVMreg	Bagging	IBk1	IBk11
rDisjunctB	0.0031	0.0046	0.0086	0.0038	0.0036	0.5240	0.5726
rDisjunctN	0.0030	0.0044	0.0087	0.0026	0.0038	0.5722	0.7740
rLinear	0.0029	0.0044	0.0081	0.0026	0.0139	0.5605	0.7780
rLinNoisy	0.0029	0.0044	0.0087	0.0026	0.0130	0.5727	0.7932
rLocLinear	0.0029	0.0044	0.0087	0.0026	0.0130	0.5729	0.7728
rNonLinPoly	0.0029	0.0046	0.0087	0.0026	0.0155	0.5667	0.7752
rNonLinTrig	0.0029	0.0045	0.0088	0.0026	0.0142	0.5494	0.7891
rRandom	0.0027	0.0041	0.0080	0.0024	0.0121	0.4506	0.6180
rRedundant	0.0029	0.0044	0.0087	0.0026	0.0039	0.7195	0.8090
rXor	0.0034	0.0050	0.0094	0.0040	0.0077	0.5369	0.6828
autoMpg	0.0051	0.0062	0.0238	0.0062	0.0055	0.1764	0.2641
bodyfat	0.0048	0.0065	0.0154	0.0048	0.0090	0.2073	0.3263
concrete	0.0035	0.0051	0.0109	0.0033	0.0125	0.4510	0.6119
elevators	0.0069	0.0087	0.0201	0.0068	0.0226	8.9453	9.9861
fishcatch	0.0039	0.0053	0.0140	0.0049	0.0072	0.1017	0.1338
fruitfly	0.0030	0.0044	0.0104	0.0039	0.0036	0.0417	0.0534
housing	0.0048	0.0063	0.0147	0.0059	0.0100	0.4033	0.5235
machinecpu	0.0031	0.0046	0.0094	0.0028	0.0077	0.0895	0.1404
pollution	0.0051	0.0067	0.0163	0.0051	0.0068	0.0620	0.1119
stock	0.0037	0.0053	0.0114	0.0035	0.0114	0.5742	0.6509
wine	0.0043	0.0060	0.0132	0.0041	0.0160	2.8720	4.1039
wisconsin	0.0102	0.0121	0.0356	0.0105	0.0106	0.3850	0.6141

Tabela A.6: Povprečne variance $\overline{\sigma^2}$.

	LinearRegression	M5P	MPerceptron	SVMreg	Bagging	IBk1	IBk11
rDisjunctB	0.1363	0.0554	0.0564	0.0738	0.0564	0.05885	0.0594
rDisjunctN	0.0582	0.0734	0.0297	0.0532	0.0796	0.11151	0.0546
rLinear	0.0557	0.0557	0.0554	0.0557	0.0472	0.05413	0.0556
rLinNoisy	0.0462	0.0463	0.0401	0.0459	0.0422	0.05548	0.0520
rLocLinear	0.0403	0.0173	0.0183	0.0402	0.0177	0.03882	0.0313
rNonLinPoly	0.0464	0.0134	0.0134	0.0339	0.0128	0.03441	0.0280
rNonLinTrig	0.0512	0.0547	0.0558	0.0522	0.0407	0.07467	0.0571
rRandom	0.0000	0.0000	0.0025	0.0768	0.0311	0.40255	0.0847
rRedundant	0.0580	0.0734	0.0351	0.0696	0.0796	0.11818	0.0738
rXor	0.0417	0.1678	0.1654	0.0417	0.1759	0.16829	0.1685
autoMpg	0.0305	0.0227	0.0094	0.0374	0.0119	0.03235	0.0266
bodyfat	0.0048	0.0047	0.0048	0.0048	0.0089	0.02606	0.0133
concrete	0.0270	0.0069	0.0092	0.0245	0.0142	0.04113	0.0337
elevators	0.0056	0.0015	0.0032	0.0041	0.0061	0.01423	0.0128
fishcatch	0.0166	0.0115	0.0151	0.0498	0.0090	0.03543	0.0283
fruitfly	0.0000	0.0000	0.0130	0.0666	0.0369	0.05828	0.0765
housing	0.0225	0.0002	0.0056	0.0123	0.0045	0.01284	0.0152
machinecpu	0.0092	0.0033	0.0022	0.0053	0.0054	0.01500	0.0164
pollution	0.0038	0.0038	0.0075	0.0125	0.0076	0.02368	0.0225
stock	0.0154	0.0077	0.0087	0.0128	0.0137	0.04710	0.0398
wine	0.0010	0.0009	0.0008	0.0015	0.0080	0.03872	0.0252
wisconsin	0.0079	0.0016	0.0045	0.0062	0.0058	0.02688	0.0244

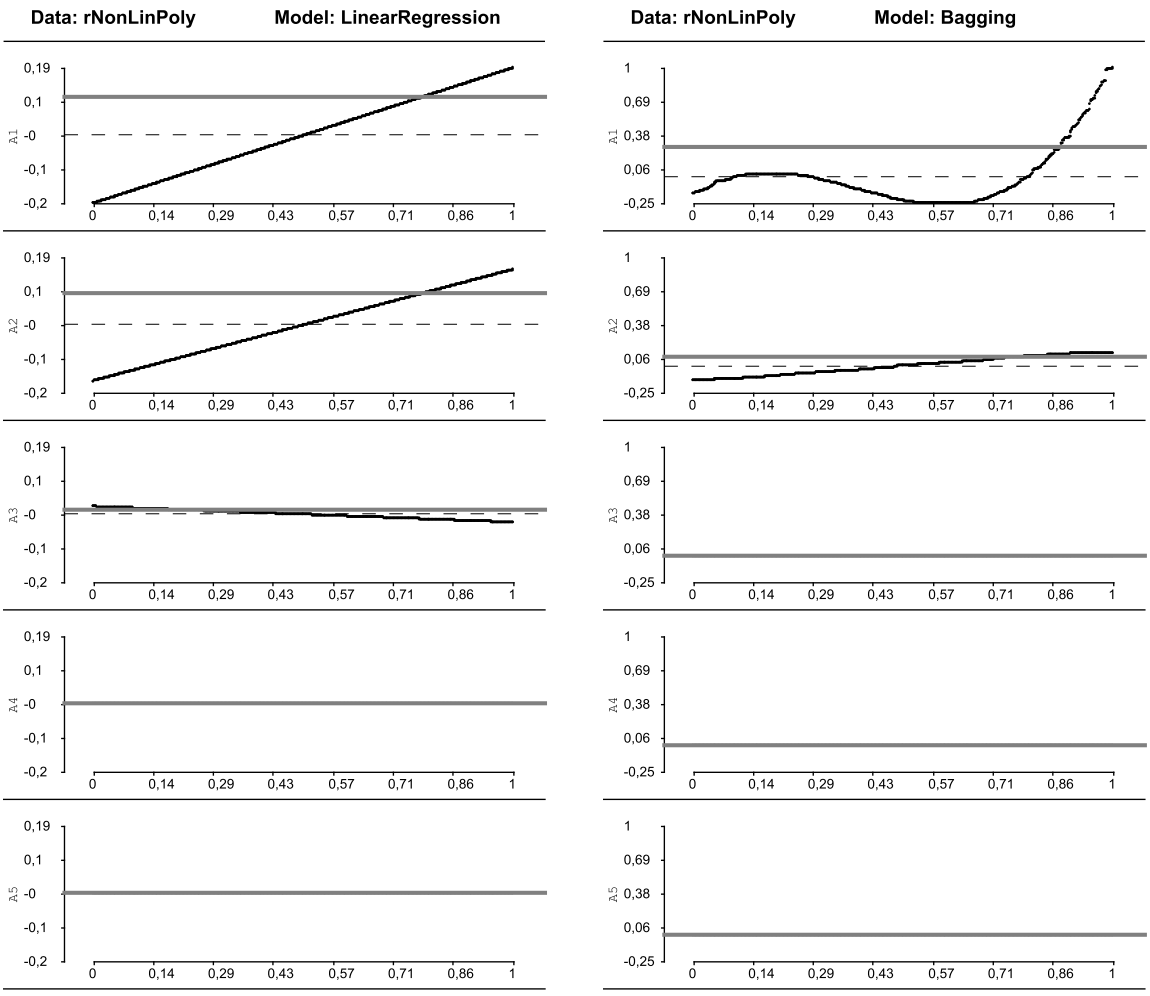
Dodatek B

Razlage modelov



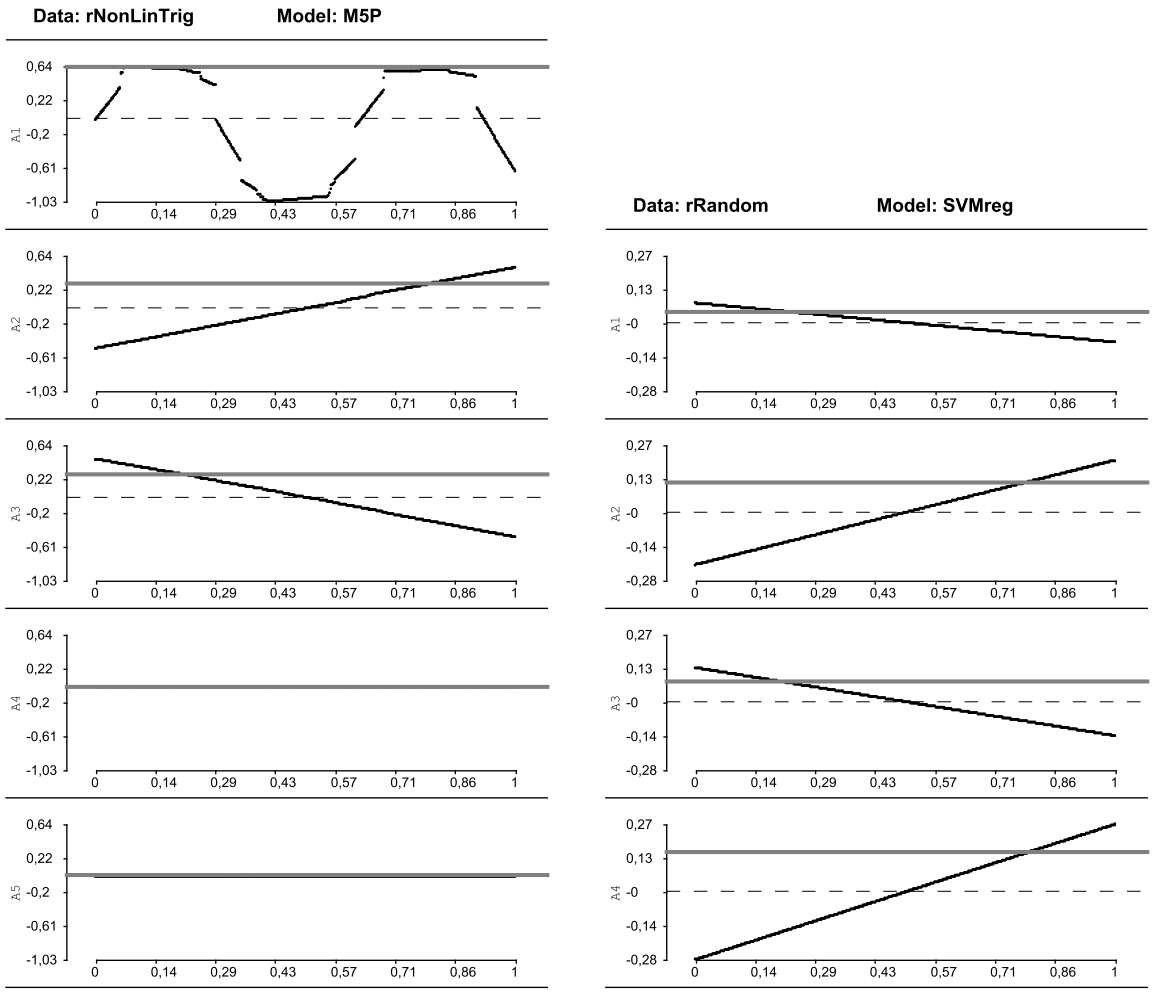
(a) (b)

Slika B.1: Množica cRedundant je podobna množici cDisjunctN. Razlika je le v tem, da sta četrti in peti atribut pri cRedundant kopiji prvega in drugega. Boosting z odločitvenimi drevesi, ki kopij ne upoštevajo, zgradi model, ki je na pogled popolnoma enak odločitvenemu drevesu za cDisjunctN (a). Če namesto dreves kot osnovni model uporabimo naivnega Bayesa, dobimo drugačne rezultate in model, ki upošteva vseh pet atributov, pri čemer so splošni prispevki vrednosti pri kopijah enaki kot pri prvem in drugem atributu (b).



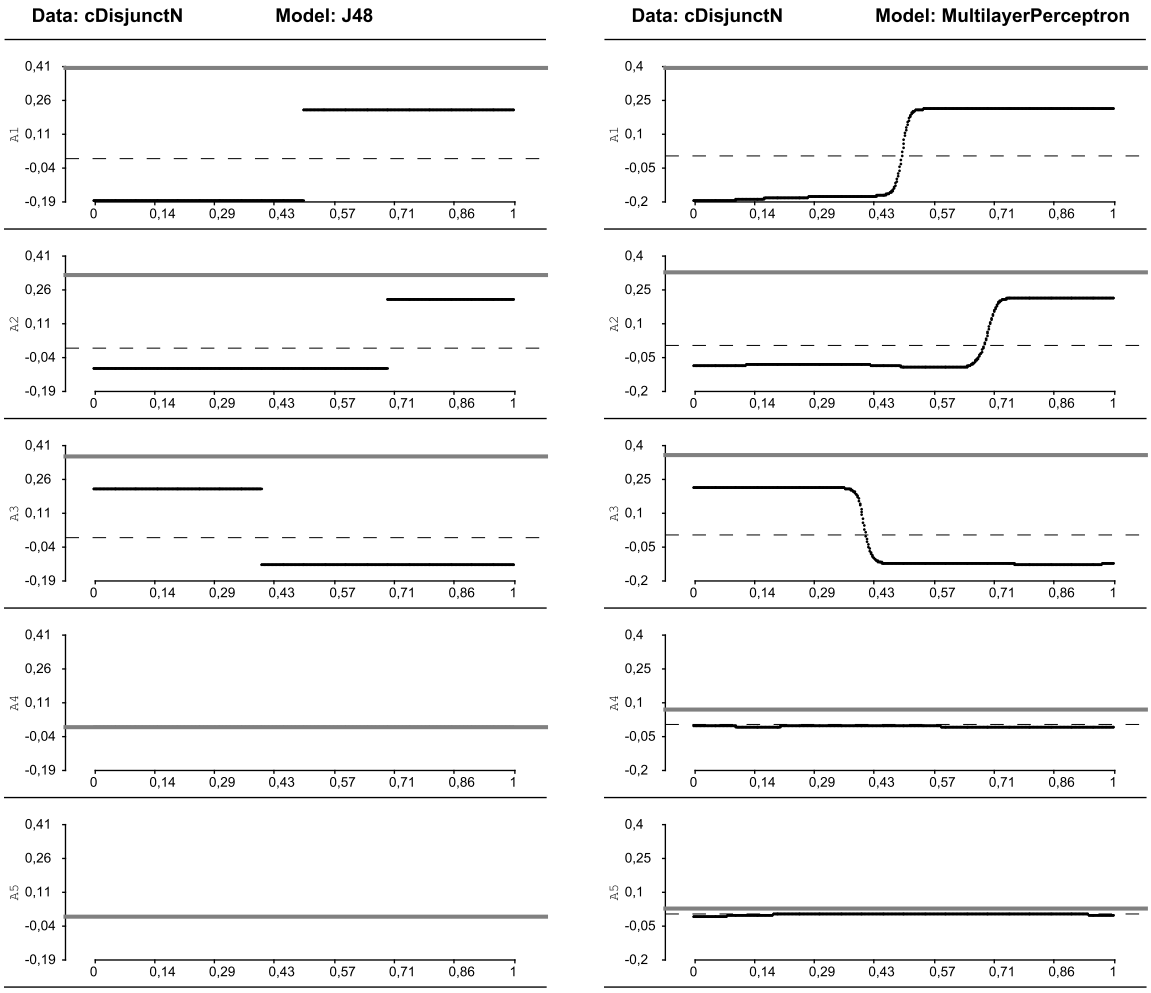
(a) (b)

Slika B.2: Na levi sliki je lepo vidna nelinearna funkcija prispevkov vrednosti prvega atributa za model bagging na množici rNonLinPoly (a). Drugi atribut ima linearno funkcijo prispevkov vrednosti in je manj pomemben, preostali trije atributi pa so nepomembni. Z osnovno linearno regresijo ne moremo modelirati polinomov druge ali višje stopnje, zato le delno opiše zvezo med prvim atributom in odvisno spremenljivko in ima temu primerno višjo napako (b).



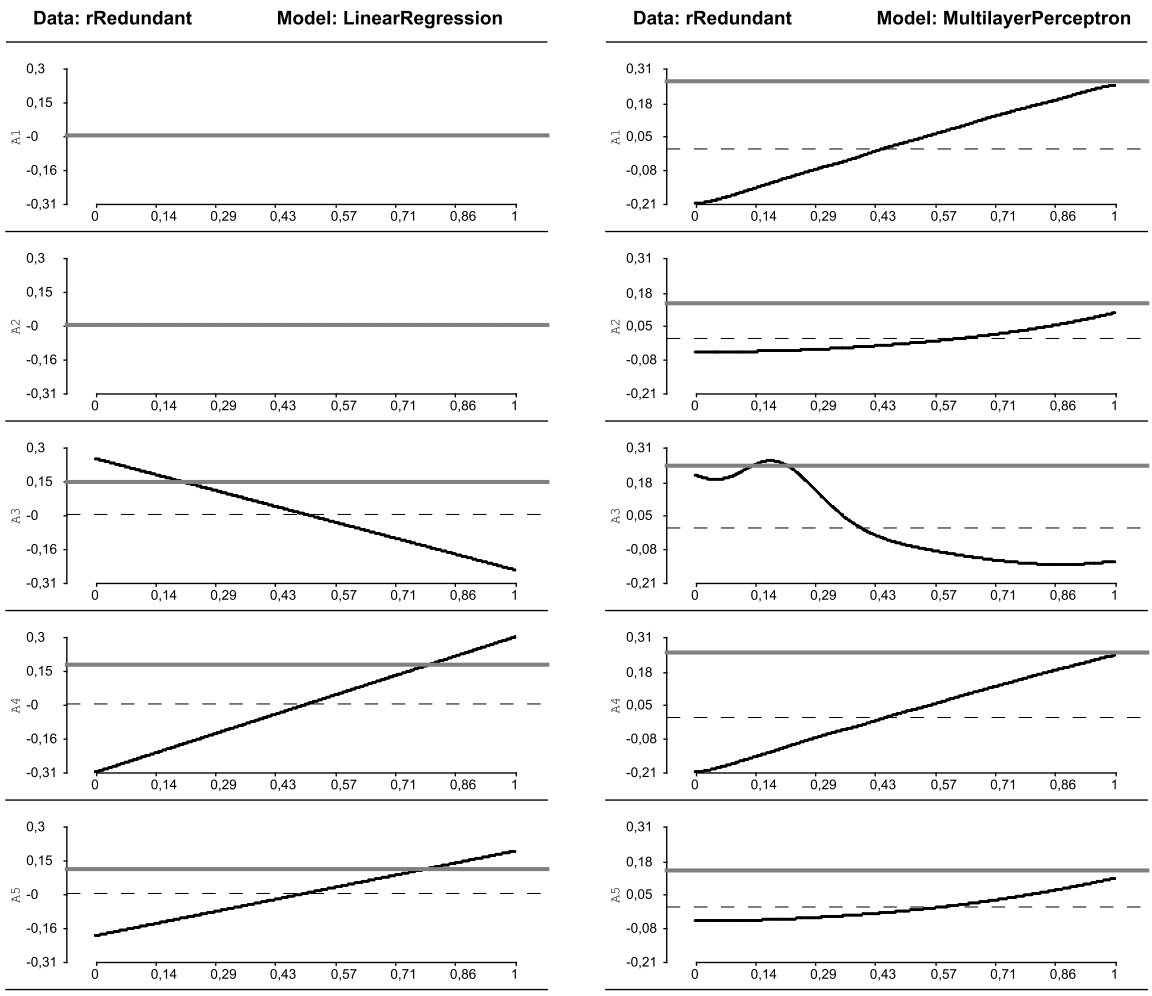
(a) (b)

Slika B.3: Na levi je vizualizacija za model regresijske drevesa, ki na za drevesa značilen način modelira trigonometrični koncept. Vizualizacija na desni pa razkrije več podrobnosti o tem, kako se je regresijski SVM preveč prilagodil podatkom iz množice rRandom. V splošnem ne moremo samo iz te vizualizacije soditi ali se je model preveč prilagodil podatkom ali ne. Potrebujemo njegovo točnost (napako) in referenčno točnost ali pa moramo poznati koncepte, ki se naj bi jih naučil. V našem primeru smo imeli slednje – vedeli smo, da so vsi štirje atributi nepomembni, zato je vsakršen vpliv posledica prevelikega prilagajanja.



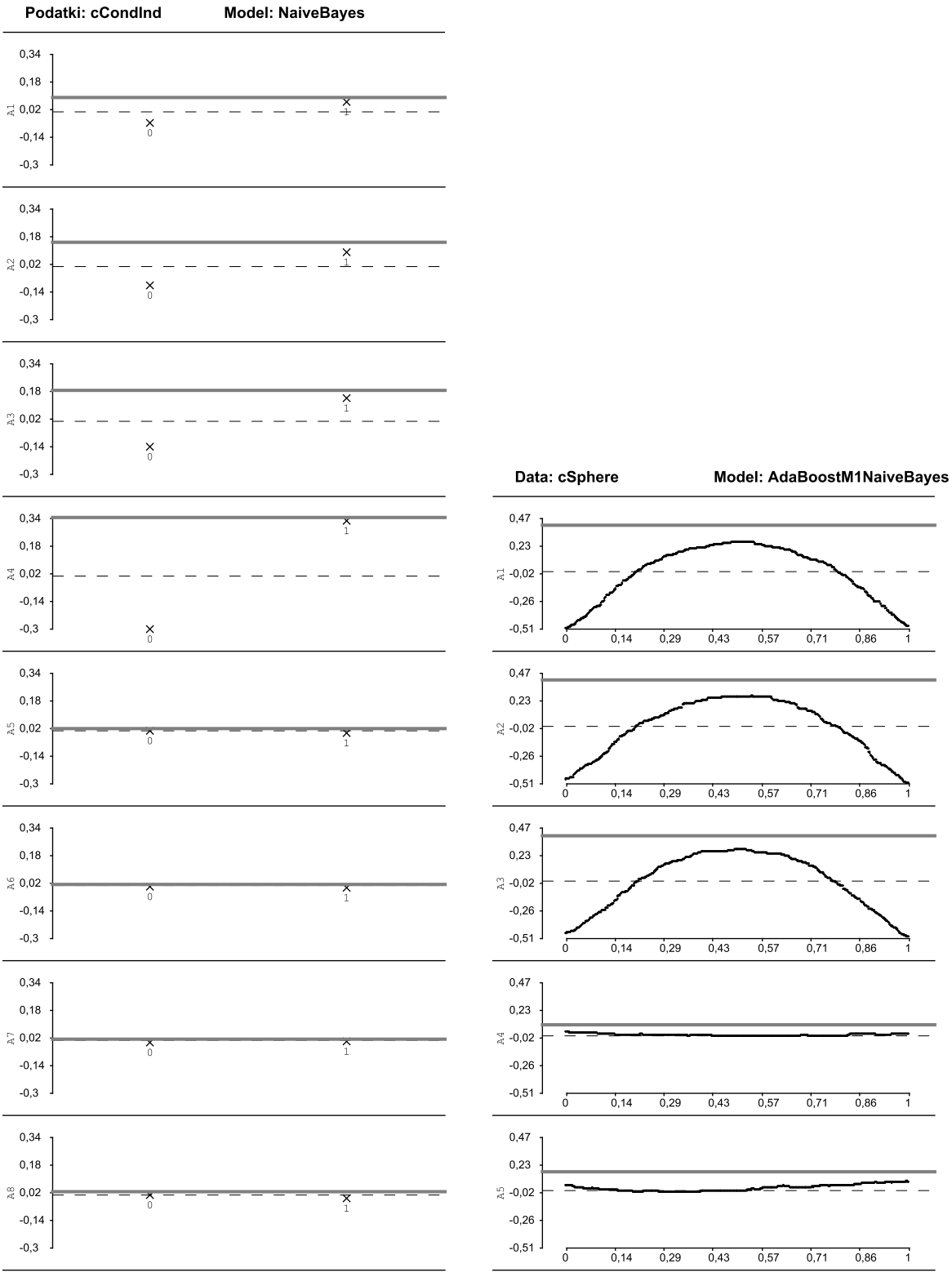
(a) (b)

Slika B.4: Funkcije splošnih prispevkov vrednosti za posamezne atribute za modela odločitvena drevesa in umetno nevronsko mrežo. Oba modela imata visoko točnost, a med njima so določene manjše razlike. Umetna nevronska mreža v manjši meri upošteva tudi oba nepomembna atributa, vidni pa so tudi zvezni prehodi prispevkov vrednosti, ki so pri odločitvenem drevesu skokoviti.



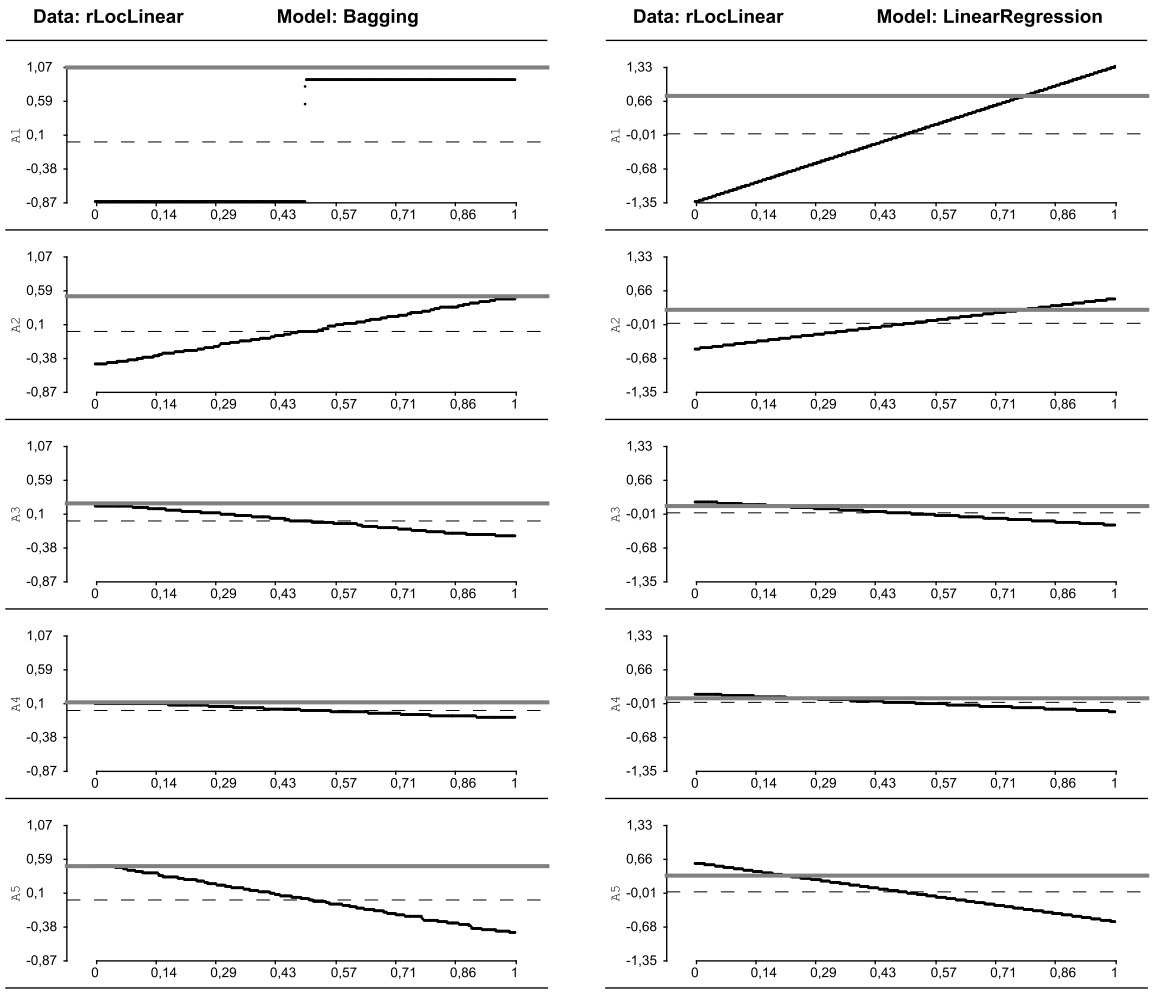
(a) (b)

Slika B.5: Linearna regresija običajno odstrani atribute, ki so močno korelirani. Umetna nevronska mreža upošteva tudi kopije atributov.



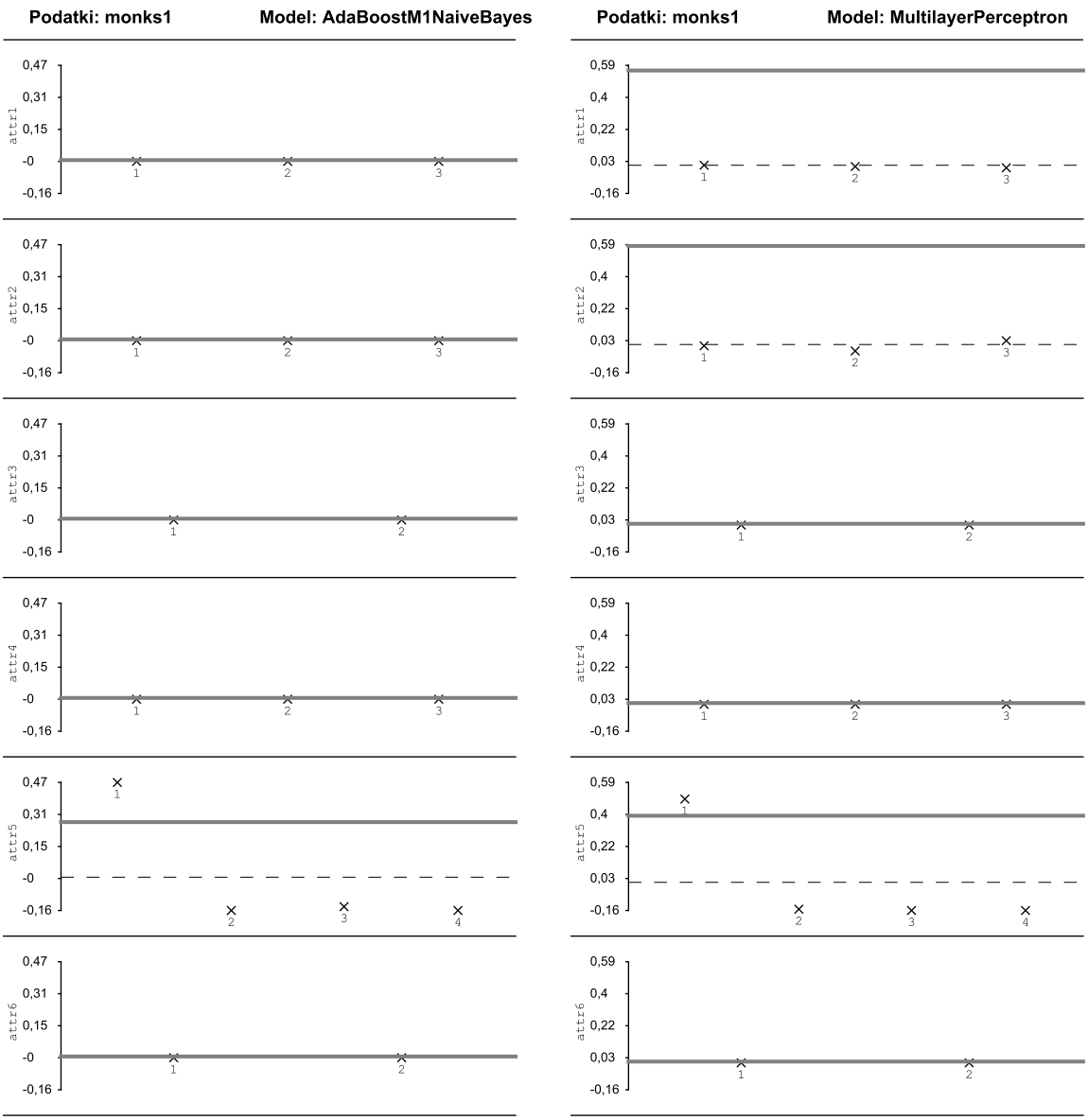
(a) (b)

Slika B.6: Naivni Bayes dobro modelira cCondInd, saj so atributi pogojno neodvisni glede na vrednost razreda. Naivni Bayes z metaučenjem bagging dobro modelira tudi podatke iz množice cSphere. Iz vizualizacij obeh modelov lahko razberemo pomembne atribute in prispevke njihovih vrednosti.



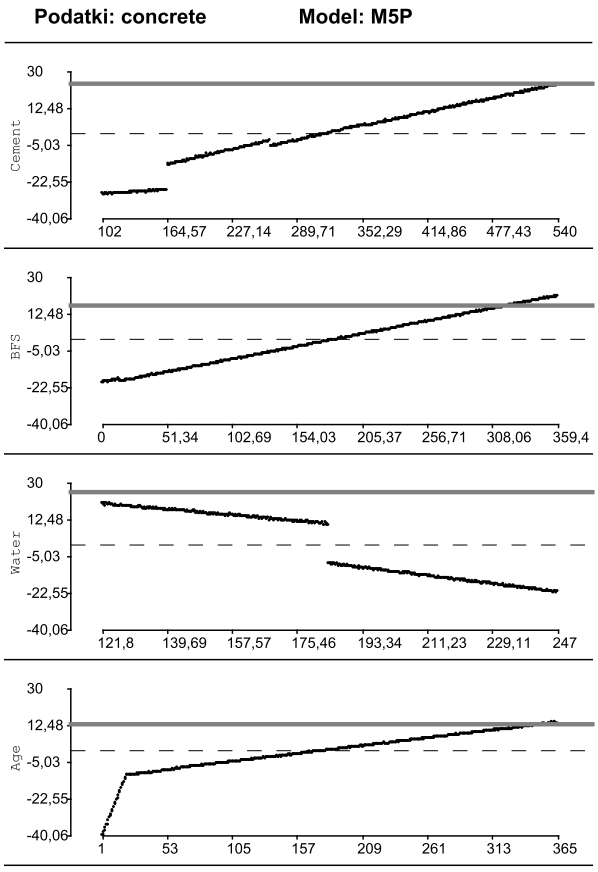
(a) (b)

Slika B.7: Problem rLocLinear je sestavljen iz dveh linearnih konceptov, med katerima izbiramo na podlagi vrednosti prvega atributa. Kot tak je pisan na kožo regresijskim drevesom, ki dosežejo najmanjšo napako, blizu jim je metaučenje bagging z regresijskimi drevesi, katerega vizualizacija je na sliki. Linearna regresija ima za ta problem zelo visoko napako, saj le približno modelira dejanske koncepte.

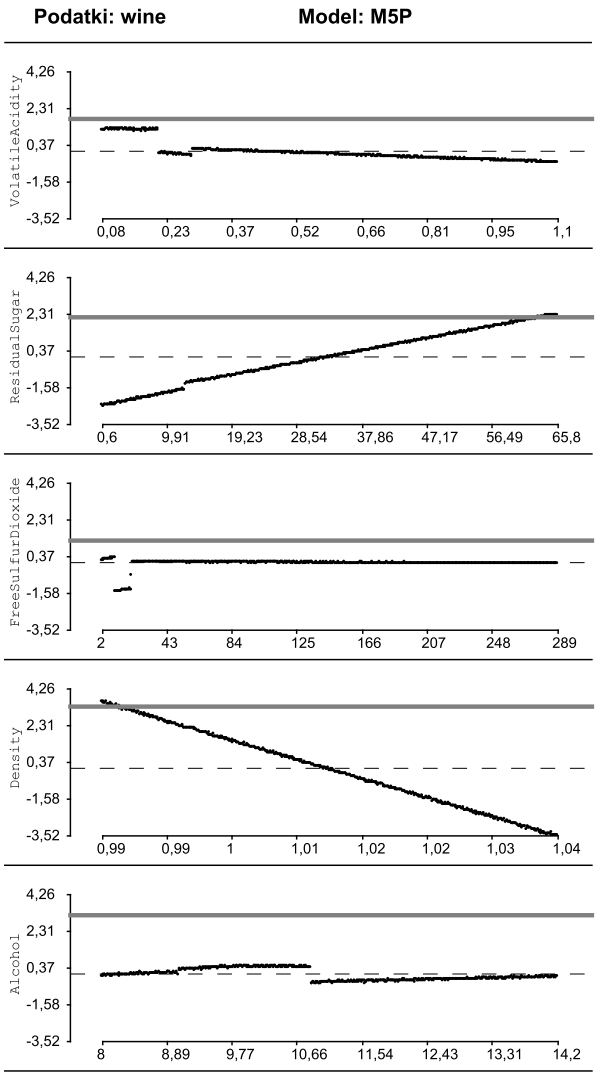


(a) (b)

Slika B.8: Če primerjamo vizualizaciji za naivnega Bayesa in umetno nevronsko mrežo vidimo razliko med obema modeloma. Naivni Bayes ne upošteva prvih dveh atributov.



Slika B.9: Vizualizaciji prispevkov štirih najpomembnejših atributov za M5P na množici concrete. Najpomembnejša atributa za določanje trdnosti betona z M5P sta vsebnost vode in cementa. Iz stroke vemo, da je pomembno razmerje med količinama teh dveh elementov mešanice betona, ampak v povprečju velja, da trdnost narašča z deležom cementa in pada z deležem vode. Nekoliko manj pomembna sta vsebnost plavžne žlindre in starost betona. Plavžna žlindra je dodatek cementu oz. betonu, ki povečuje trdnost betona, kar je razvidno tudi iz vizualizacije. Prispevek starosti ni povsem linearen, kar je razumljivo, saj se beton na začetku krepi hitreje.



Slika B.10: Regresijsko drevo na množici wine doseže najnižjo napako. Glede na pomembnost si sledijo atributi Density, Alcohol, ResidualSugar, VolatileAcidity in FreeSulfurDioxide. V relevantni literaturi [19] najdemo nekoliko drugačen vrstni red: Alcohol, Sulphates, pH, FreeSulfurDioxide, VolatileAcidity. Navedeni vrstni red je pomembnost atributov za model SVM, za katerega avtorji navedejo točnost 64.3%. Ker so avtorji problem obravnavali kot klasifikacijski in ne regresijski, smo poskus ponovili. Če problem obravnavamo kot klasifikacijski, potem Boosting z odločitvenim drevesom J48 doseže primerljivo točnost (65.7%), vrstni red atributov pa se spremeni v Alcohol, VolatileAcidity, Sulphates, pH in FreeSulfurDioxide.

Dodatek C

Vprašalnika za preverjanje uporabnosti razlage

Povprečni predstavnik določene vrste murnov oz. čričkov (Oecanthus fultoni) se v običajnih pogojih oglasi približno 112-krat na minuto. Nekateri predstavniki se oglašajo pogosteje, nekateri pa redkeje.

Na pogostost oglašanja murna morda vplivajo sledeči dejavniki: dolžina murna, temperatura zraka, vlaga. Spodaj so podatki o sedmih različnih murnih, za katere smo izmerili pogostost oglašanja pod različnimi pogoji:

Muren št. 1	dolžina [mm]	vlaga [%]	temp [°C]	vrednost za murna [oglašanj / min]
[redacted]	11.76	70	12	52
Povprečje=112	0	0	-60	

Muren št. 2	dolžina [mm]	vlaga [%]	temp [°C]	vrednost za murna [oglašanj / min]
[redacted]	11.56	55	16	82
Povprečje=112	0	0	-30	

Muren št. 3	dolžina [mm]	vlaga [%]	temp [°C]	vrednost za murna [oglašanj / min]
[redacted]	11.86	32	22	127
Povprečje=112	0	0	+15	

Muren št. 4	dolžina [mm]	vlaga [%]	temp [°C]	vrednost za murna [oglašanj / min]
[redacted]	12.12	80	12	60
Povprečje=112	0	0	-52	

Muren št. 5	dolžina [mm]	vlaga [%]	temp [°C]	vrednost za murna [oglašanj / min]
[redacted]	12.31	74	28	172
Povprečje=112	0	0	+60	

Muren št. 6	dolžina [mm]	vlaga [%]	temp [°C]	vrednost za murna [oglašanj / min]
[redacted]	11.65	44	21	120
Povprečje=112	0	0	+8	

Muren št. 7	dolžina [mm]	vlaga [%]	temp [°C]	vrednost za murna [oglašanj / min]
[redacted]	11.34	72	25	150
Povprečje=112	0	0	+38	

Poleg izmerjenih vrednosti so pri vsakem murnu v pomoč tudi prispevki posameznih dejavnikov k pogostosti oglašanja. Poglejmo si primer št. 1: Povprečen predstavnik se oglasi 112-krat na minuto, a ta muren se je oglašal le 52-krat. Celotno razliko smo pripisali temperaturi 12°C, ki torej zmanjša število oglašanj na minuto. Dolžina in vlaga v tem primeru ne vplivata na pogostost oglašanja (prispevek je pri obeh enak 0).

Iz zgornjih primerov poskušaj razbrati, kateri izmed treh dejavnikov vplivajo na pogostost oglašanja (in na kakšen način). Na podlagi ugotovitev poskušaj napovedati pogostost oglašanja za štiri druge murne. Pri vsakem označi tudi stopnjo prepričanosti v pravilnost napovedi:

Muren št. 8	dolžina [mm]	vlaga [%]	temp [°C]	vrednost za murna [oglašanj / min]
[redacted]	11.91	59	21	?

Prepričanost v lastno napoved za ta primer (obkroži):
zelo neprepričan neprepričan prepričan zelo prepričan

Muren št. 10	dolžina [mm]	vlaga [%]	temp [°C]	vrednost za murna [oglašanj / min]
[redacted]	11.87	28	29	?

Prepričanost v lastno napoved za ta primer (obkroži):
zelo neprepričan neprepričan prepričan zelo prepričan

Muren št. 9	dolžina [mm]	vlaga [%]	temp [°C]	vrednost za murna [oglašanj / min]
[redacted]	12.00	54	27	?

Prepričanost v lastno napoved za ta primer (obkroži):
zelo neprepričan neprepričan prepričan zelo prepričan

Muren št. 11	dolžina [mm]	vlaga [%]	temp [°C]	vrednost za murna [oglašanj / min]
[redacted]	11.73	33	17	?

Prepričanost v lastno napoved za ta primer (obkroži):
zelo neprepričan neprepričan prepričan zelo prepričan

B2

Slika C.1: Vprašalnik za preverjanje uporabnosti razlage pri napovedovanju frekvence oglašanja murna. Pravilne napovedi so 120, 180, 165 in 90.

Nek proizvajalec insekticidov je testiral učinkovitost različnih vrst insekticidov. Na učinkovitost aplikacije insekticida (delež odpravljenih insektov) morda vplivajo sledeči dejavniki: tip insekticida (A, B ali C), količina uporabljenega insekticida, temperatura zraka.

Spodaj so podatki o sedmih različnih testih, v katerih so izmerili učinkovitost pod različnimi pogoji:

Test št. 1	tip [A,B,C]	količina [ml]	temp [°C]	učinkovitost [%]
[redacted]	A	450	21	27
Povprečje=47	-31	+11	0	

Test št. 4	tip [A,B,C]	količina [ml]	temp [°C]	učinkovitost [%]
[redacted]	A	250	21	2
Povprečje=47	-25	-20	0	

Test št. 2	tip [A,B,C]	količina [ml]	temp [°C]	učinkovitost [%]
[redacted]	B	280	22	12
Povprečje=47	-7	-28	0	

Test št. 3	tip [A,B,C]	količina [ml]	temp [°C]	učinkovitost [%]
[redacted]	B	600	20	70
Povprečje=47	+3	+20	0	

Test št. 5	tip [A,B,C]	količina [ml]	temp [°C]	učinkovitost [%]
[redacted]	B	800	22	75
Povprečje=47	+4	+24	0	

Test št. 6	tip [A,B,C]	količina [ml]	temp [°C]	učinkovitost [%]
[redacted]	C	250	21	56
Povprečje=47	+31	-22	0	

Test št. 7	tip [A,B,C]	količina [ml]	temp [°C]	učinkovitost [%]
[redacted]	C	200	19	40
Povprečje=47	+28	-35	0	

Poleg izmerjenih vrednosti so pri vsakem testu v pomoč tudi prispevki posameznih dejavnikov k učinkovitosti. Poglejmo si primer št. 1: Povprečna izmerjena učinkovitost pri teh testih znaša 47%, a pri tem testu je bila le 27% . Razliko smo pripisali uporabljenemu tipu insekticida (A), ki govori v prid zmanjšani učinkovitosti, in uporabljeni količini, ki govori v prid večji učinkovitosti. Temperatura v tem primeru ne vpliva na učinkovitost (prispevek je enak 0).

Iz zgornjih primerov poskušaj razbrati, kateri izmed treh dejavnikov vplivajo na učinkovitost aplikacije insekticida (in na kakšen način). Na podlagi ugotovitev poskušaj napovedati učinkovitost za štiri druge teste. Pri vsakem označi tudi stopnjo prepričanosti v pravilnost napovedi:

Test št. 8	tip [A,B,C]	količina [ml]	temp [°C]	učinkovitost [%]
[redacted]	A	600	21	?

Prepričanost v lastno napoved za ta primer (obkroži):
zelo neprepričan neprepričan prepričan zelo prepričan

Test št. 10	tip [A,B,C]	količina [ml]	temp [°C]	učinkovitost [%]
[redacted]	C	400	20	?

Prepričanost v lastno napoved za ta primer (obkroži):
zelo neprepričan neprepričan prepričan zelo prepričan

Test št. 9	tip [A,B,C]	količina [ml]	temp [°C]	učinkovitost [%]
[redacted]	B	100	23	?

Prepričanost v lastno napoved za ta primer (obkroži):
zelo neprepričan neprepričan prepričan zelo prepričan

Test št. 11	tip [A,B,C]	količina [ml]	temp [°C]	učinkovitost [%]
[redacted]	C	800	20	?

Prepričanost v lastno napoved za ta primer (obkroži):
zelo neprepričan neprepričan prepričan zelo prepričan

B

Slika C.2: Vprašalnik za preverjanje uporabnosti razlage pri napovedovanju učinkovitosti insekticida. Pravilne napovedi so 30, 94, 2 in 100.