UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Julian Klauser

Razlaga napovedi regresijskih modelov s prispevki posameznih vhodnih spremenljivk

DIPLOMSKO DELO NA UNIVERZITETNEM ŠTUDIJU

MENTOR
prof. dr. Igor Kononenko

Ljubljana, 2010

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01597/2009
Datum: 15.10.2009

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **JULIAN KLAUSER**

Naslov: **RAZLAGA NAPOVEDI REGRESIJSKIH MODELOV S PRISPEVKI POSAMEZNIH VHODNIH SPREMENLJIVK**

EXPLAINING REGRESION PREDICTION WITH CONTRIBUTIONS OF INPUT VARIABLES

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Razlaga posameznih napovedi modelov je zelo pomemben vidik odkrivanja zakonitosti v podatkih, saj se ukvarja s premostitvijo vrzeli med rezultati strojnega učenja, statistične analize in končnim uporabnikom. Z razširitvijo obstoječe metode za splošno razlago klasifikacijskih modelov, ki jo je razvil Erik Štrumbelj, bo naloga implementirali metodo za razlago regresijskih modelov in razrešiti probleme, ki so značilni za regresijo. Učinkovitost in uspešnost metode bo potrebno preveriti na znanih množicah podatkov in množicah podatkov, ki bodo zasnovane posebej v ta namen. V analizi rezultatov bo potrebno primerjati razlage različnih regresorjev med seboj glede na uspešnost posameznih regresorjev v posameznih domenah.

Mentor:

prof. dr. Igor Kononenko

Dekan:

prof. dr. Franc Solina

Fakulteta za računalništvo in informatiko
2
UNIVERZA V LJUBLJANI

Zahvala

Rad bi se zahvalil svojemu mentorju, prof. Igorju Kononenku, ter mlademu raziskovalcu Eriku Štrumblju za vso pomoč in nasvete. Hvala tudi staršem, bratu in punci za potrpežljivost in podporo.

Kazalo

Kratice

SVM	support vector machine (metoda podpornih vektorjev)
KNN	*k*-nearest neighbors algorithm (algoritem *k*-najbližjih sosedov)
M5P	regression tree algorithm for M5 Model trees (algoritem regresijskih dreves za M5 drevesa)
MP	multilayer perceptron [artificial neural network] (večnivojski perceptron [umetna nevronska mreža])
LR	linear regression (linearna regresija)
RMSE	root mean squared error (koren srednje kvadratne napake)
RRMSE	relative root mean squared error (relativni koren srednje kvadratne napake)

Povzetek

Razlaga napovedi regresijskih algoritmov je pomemben vidik odkrivanja zakonitosti v podatkih. Metodo za razlago v klasifikaciji, ki sta jo razvila Štrumbelj in Kononenko, v tej diplomski nalogi razširimo na regresijo. Osnova metode izhaja iz teorije iger in s pomočjo vzorčenja reši problem eksponentne časovne zahtevnosti. Predstavili smo delovanje metode in jo po uspešni implementaciji preizkusili z različnimi regresijskimi algoritmi na umetnih domenah. Analizirali smo uspešnost in učinkovitost metode ter primerjali algoritme glede na uspešnost napovedi ter uspešnost razlage. Poleg obstoječe vizualizacije razlag napovedi smo realizirali nov način vizualizacije razlage posameznih napovedovalnih modelov. Pokazali smo tudi način, kako lahko metodo razširimo, da upošteva tudi kontekst z delnim upoštevanjem porazdelitve vrednosti atributov.

Ključne besede

strojno učenje, razlaga napovedi, regresija, vizualizacija razlage, vizualizacija modela, prispevek atributa, razlaga v kontekstu.

Abstract

The explanation of predictions, made by regression algorithms, is a vey important aspect of the discovery of patterns in data. In this thesis we extend an existing method for explanation in classification, developed by Štrumbelj and Kononenko, to regression. The base of the method comes from game theory and with the help of sampling, solves the problem of exponential time growth. We presented the basic concept of the method and after its successful implementation, we tested it with various regression algorithms on artificial domains. We analyzed the performance and efficiency of the presented method and compared algorithms according to the success of the prediction and the success of the explanation. Besides the existing visualisation of the prediction's explanation we created a new way of visualising the explanation models. We have also shown that our method can be extended, so that it is able to cleave to the context with partial compliance to attribute value distribution.

Keywords

machine learning, prediction explanation, regression, explanation visualisation, model visualisation, attribute contribution, explanation in context.

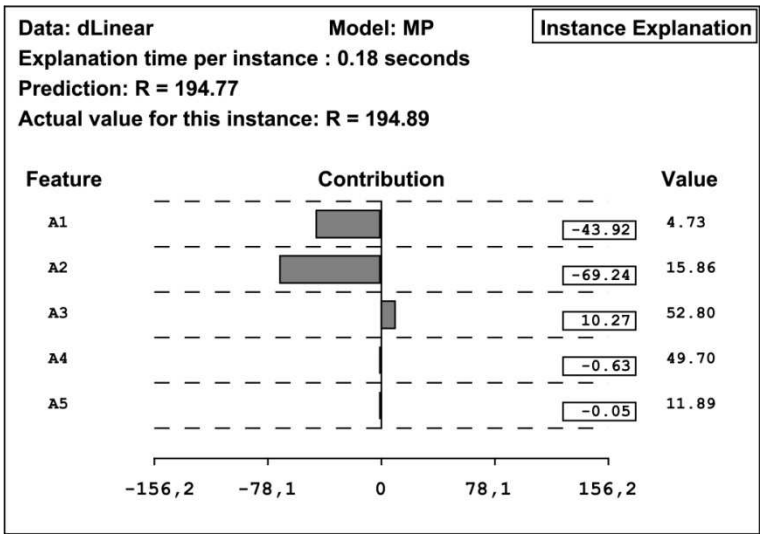
1 UVOD

Ena izmed glavnih nalog strojnega učenja je prepoznavanje kompleksnih struktur s pomočjo množice podatkov ter z inteligentnimi napovedmi pomagati človeku reševati različne probleme. Prav tako pomembna je razlaga teh napovedi, da z njimi lažje posredujemo znanje, pridobljeno iz napovedi. S pomočjo razlage napovedi lažje ocenimo pomembnost attributa ter njegovo vlogo pri napovedi. Izbira vrste razlage in izračun le-te ni trivialen problem. Velika raznolikost med napovedovalnimi algoritmi privede do tega, da bi bodisi potrebovali za vsak algoritem svojo razlago (kar zaplete proces in naredi razlago nepregledno, še posbej za običajnega uporabnika), bodisi uporabili obstoječe splošne razlage (ki pa niso uspešne na vseh problemih). V tej diplomski nalogi bomo preizkusili metodo, ki naj bi delovala neodvisno od napovedovalnega algoritma (tako imenovani "blackbox" princip), a kljub temu učinkovito. To nam omogoča enoviten način za razlago velike množice napovedovalnih algoritmov in s tem olajša interpretacijo rezultatov ter primerjavo med različnimi algoritmi.

Glavna inspiracija za nalogo prihaja iz članka [2], v katerem je opisana metoda za razlago napovedi regresijskih algoritmov s pomočjo numeričnega odvoda. Vendar pa je metoda vsebovala tri večje pomankljivosti. Prva težava je bila, da regresijsko drevo (kot primer družine modelov, na katerih metoda ne deluje dobro) zgradi stopničasto funkcijo, kjer je numerični odvod enak nič. Težavo sicer odpravitmo s povezavo stopnic, vendar je natančnost napovedi precej nižja in je bila zelo odvisna od oblike stopničaste funkcije. Druga težava je bila z diskretnimi atributi, katerim ne moramo določiti ustrezne številčne vrednosti in tako ni mogoče izračunati velikost spremembe atributa in posledično odvoda. Tretja težava pa je bila lokalnost razlage. Razlog zanjo tiči že v definiciji numeričnega odvoda, saj le ta obravnava neposredno bližino trenutne vrednosti in je zato metoda precej kratkovidna. Te pomankljivosti smo poskusili odpraviti z novim načinom razlage, ki temelji na obstoječi metodi za splošno razlago klasifikacijskih modelov, ki sta jo razvila Štrumbelj in Kononenko [1]. V članku [1] je opisan učinkovit algoritem, ki s kombinacijo teorije iger ter vzorčenja uspešno razlaga klasifikacijske odločitve. Metodo smo prilagodili na regresijo in testirali uspešnost razlage med posameznimi napovedovalnimi algoritmi.

Za lažjo predstavo o tem, kaj smo želeli doseči, si poglejmo preprost primer. Na Sliki 1 je prikazana razlaga za preprost umetni problem. Regresijska spremenljvika R je v podatkih izračunana kot linearna kombinacija prvih treh atributov. Razlago generiramo tako, da razliko

napovedi in povprečne vrednosti regresijske spremenljivke porazdelimo po atributih. Velikost prispevka atributa nam tako pove, glede na razliko od povprečne vrednosti atributa, koliko prispeva k trenutni napovedi. Če je prispevek atributa pozitiven, to kaže na večjo vrednost regresijske spremenljivke zaradi tega atributa. Obratno velja, če je prispevek atributa negativen, tedaj je vrednost regresijske spremenljivke zaradi atributa manjša. Za naš konkreten primer je povprečna vrednost regresijske spremenljivke 300, povprečna vrednost posameznega atributa pa je 50 (manjša odstopanja nastajajo zaradi zaokrožitev rezultatov).



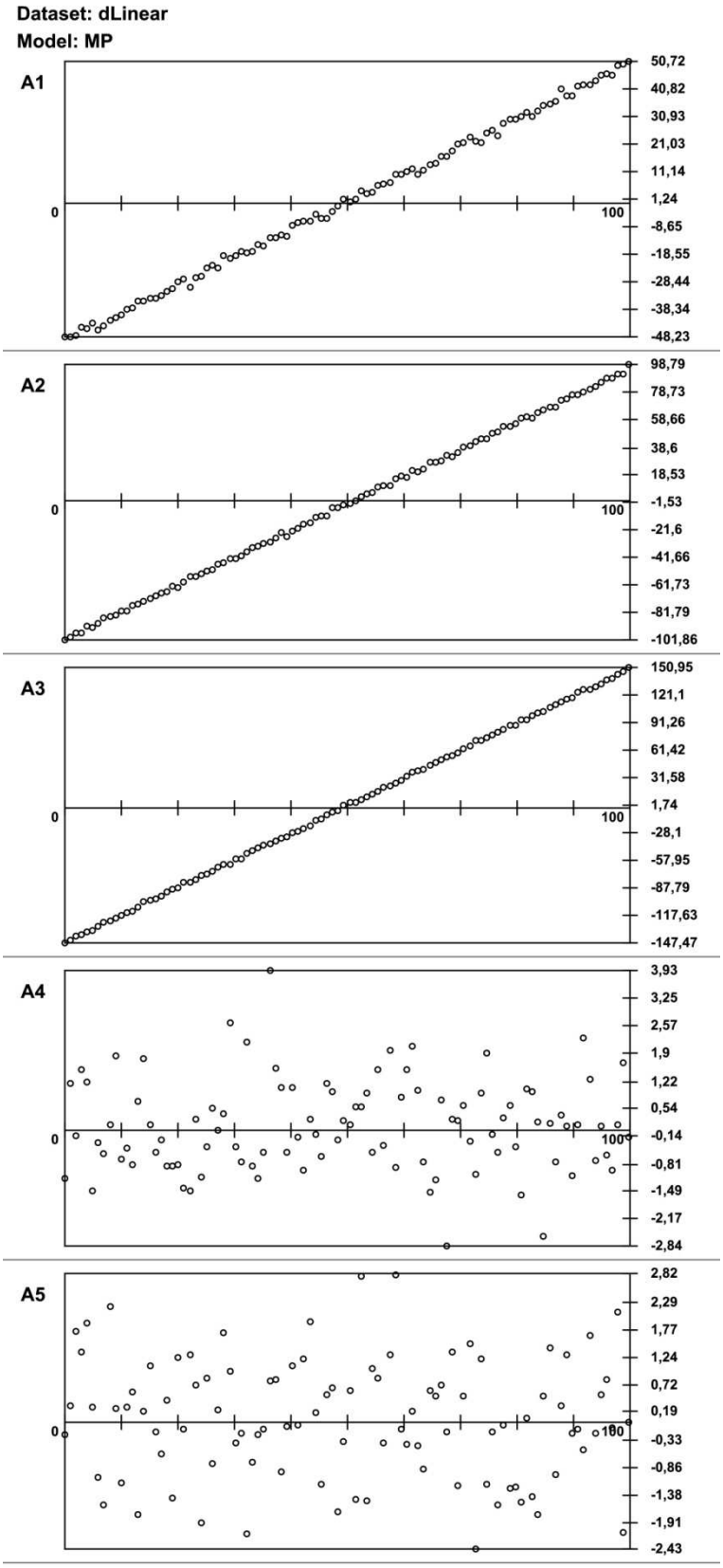
Slika 1: Preprost primer razlage napovedi nevronske mreže na podatkovni domeni linearne funkcije (R = A1 + 2*A2 + 3*A3). Atributa A4 in A5 ne vplivata (bistveno) na vrednost regresijske spremenljivke.

Ko enkrat znamo razlagati napovedi, lahko oblikujemo tudi razlage modelov. Le ta nam pove, kako se model obnaša preko celotnega spektra vrednosti posameznega atributa. Slika 2 nam kaže, da je model pravilno ugotovil, da sta atributa A4 in A5 nepomembna, ter, da prvi trije atributi linearno naraščajo. Pozoren je treba biti na dejstvo, da so skale atributov prilagojene na atribut. Torej prvi atribut ima prispevke med -50 ter 50, drugi med -100 in 100 ter tretji med -150 in 150. Tako vidimo, da prispevek atributa A1 narašča linearno s faktorjem 1, atributov A2 in A3 pa s faktorjema 2 in 3 (točno taki faktorji torej, kot so v formuli, ki je generirala podatke). Torej se je model pravilno naučil faktorje linarne funkcije, ki je služila kot generator podatkov.

Razlaga ni omejena na linearne oz. linearno aditivne modele. To so modeli, pri katerih je učinek neke spremenljivke neodvisen od vrednosti ostalih vhodnih spremenljivk (primer takega modela je, na primer, naivni Bayes). Razlaga uspešno razdeli prispevke tudi med spremenljivke, ki so v nelinearni zvezi z izhodom modela.

Tu smo opisali preprost umeten problem, vendar na enak način rešujemo praktične probleme v medicini (napoved modela je diagnoza, simptomi so opisani z atributi), trženju (model napoveduje popularnost izdelka, atributi opisujejo lastnosti izdelka) ali pa pri napovedovanju vremena (napoved modela je tu količina padavin, z atributi opišemo meteorološko stanje vremena). Podobnih praktičnih problemov lahko najdemo vsak dan več.

Vsebina naloge je organizirana na sledeč način. V naslednjem poglavju je podan formalni opis metode ter način razlage celotnega modela. V 3. poglavju opisujemo regresijske modele ter umetne podatkovne domene, ki smo jih uporabili pri testiranju metode. Poleg tega opišemo implementacijo metode ter podamo analizo rezultatov. Razdelek 3.5 opisuje razlago v kontekstu, ki predstavlja pomembno lastnost razlagalne metode in sicer, da je razlaga napovedi odvisna od konteksta podatkovne domene. V zaključnem 4. poglavju povzamemo pomembnejše ugotovitve in predstavimo možnosti nadaljnjih raziskav.



Slika 2: Razlaga modela nevronske mreže nad linearno domeno (R = A1 + 2*A2 + 3*A3). Model se je pravilno naučil faktorje linarne funkcije ter upošteval relativno nepomembnost atributov A4 in A5.

2 TEORETIČNI DEL

2.1 Regresija

Določanje vrednosti regresijske spremenljivke na podlagi vrednosti atributov je glavna naloga regresijskega prediktorja. Prostor atributov $\mathcal{A}$ opišemo z množico atributov, ki so lahko bodisi zvezni bodisi diskretni. V danem primeru vsak atribut zavzame natanko eno izmed vrednosti. Regresijska spremenljvika pa je, za razliko od klasifikacijskega razreda, zvezna. Regresijo lahko abstrahiramo na nivo funkcije. Ta funkcija slika iz prostora atributov (neodvisne spremenljivke) na realno os (odvisna spremenljivka).

2.2 Opis razlagalne metode

Naša naloga je, da razložimo, kako trenutne vrednosti atributov prispevajo k razliki v napovedi, ki jo nato razdelimo med podmnožice preko interakcij. Naj nam množica $N = \{1,2,...,n\}$ predstavlja indekse n atributov, $\mathcal{A}_N$ naj bo prostor vseh atributov množice N, f naj bo regresijska funkcija, ki predstavlja naš model, ter $x = (x_1, x_2, \ldots x_n) \in \mathcal{A}_N$ primer iz prostora atributov. Razlika napovedi $\Delta(S)$, ko poznamo le neko podmnožico atributov v S ($S \subseteq N$), je definirana kot:

$$\Delta(S) = \frac{1}{|\mathcal{A}_{N\setminus S}|} \sum_{y \in \mathcal{A}_{N\setminus S}} f\big(\tau(x, y, S)\big) - \frac{1}{|\mathcal{A}_N|} \sum_{y \in \mathcal{A}_N} f(y) \tag{1}$$

$$\tau(x, y, S) = (z_1, z_2, \ldots, z_n),\ z_n = \begin{cases} x_i\ ;\ i \in S \\ y_i\ ;\ i \notin S \end{cases}$$

Izraz $\Delta(S)$ opisuje razliko napovedi med pričakovano napovedjo, ko poznamo samo vrednosti atributov primera x, ki so vsebovane v množici S, ter pričakovano napovedjo, ko ne poznamo vrednosti atributov. Naš cilj je razdeliti $\Delta(S)$ med posamezne atribute. K tej razliki prispevajo ne samo posamezne vrednosti, ampak tudi interakcije med njimi, zato zapišemo $\Delta(S) = \sum_{W \subseteq S} I(W)$. Nova funkcija I nam opisuje interakcije vplivov vrednosti atributov. Če privzamemo, da velja za $I(\emptyset) = 0$, oblikujemo rekurzivno definicijo:

$$I(S) = \Delta(S) - \sum_{W \subset S} I(W), \qquad S \subseteq N \tag{2}$$

V naslednjem koraku razdelimo interakcije med atribute s funkcijo prispevkov atributne vrednosti $\varphi_i(\Delta)$. To storimo tako, da vsakemu vpletenemu atributu dodelimo enak delež interakcije, saj brez katerega koli atributa ta interakcija ne bi obstajala:

$$\varphi_i(\Delta) = \sum_{W \subseteq N \setminus \{i\}} \frac{I(W \cup \{i\})}{|W \cup \{i\}|}, \qquad i = 1,2,\dots,n \tag{3}$$

Pristop ima svojo analogijo v teoriji iger in Shapleyevi vrednosti, kjer imamo koalicijsko igro N igralcev (glej [1]). Podmnožice igralcev predstavljajo koalicije in karakteristična funkcija Δ predstavlja vrednost posamezne koalicije (vrednost prazne koalicije je enaka 0). Cilj je razdeliti skupni "dobiček" med igralce na podlagi njihove potencialne uspešnosti na podmnožicah, ki jo opisuje karakteristična funkcija. Najbolj razširjena rešitev je Shapleyeva vrednost. Shapleyeva vrednost ima nekaj zelo zaželjenih lastnosti. Prva taka lastnost je simetričnost: če imata dva atributa simetričen učinek preko vseh podmnožic preostalih atributov, potem dobita enako velik prispevek. Druga lastnost je učinkovitost: vsota vseh prispevkov naj bo enaka vrednosti, ki jo razlagamo.

Tretja lastnost opisuje nepomembne atribute: če atribut ne povzroči nobene spremembe preko vseh podmnožic preostalih atributov, potem naj ima prispevek enak 0. Gre za zelo željene lastnosti, saj nočemo da bi dvema identičnima atributoma dodelila različen prispevek. Kot tudi si ne želimo poljubne vsote prispevkov ali nepomembnih atributov z neničelnimi prispevki. Shapley je pokazal, da s temi tremi lastnostmi ter aditivnostjo omejimo vse rešitve na eno samo - Shapleyevo vrednost.

Žal pa je osnovni pristop časovno zelo zahteven. Pregledati je potrebno 2^n podmnožic. Da bi skrčili časovno zahtevnost, se poslužimo vzorčenja. S tem je časovna zahtevnost celotnega algoritma določena s produktom števila vzorcev, števila atributov ter hitrosti napovedovanja regresijske spremenljivke napovedovalnih algoritmov.

Na naš problem torej lahko gledamo kot na koalicijsko igro med N atributi, med katere razdelimo razliko v prispevku. Naj bo $\pi(N)$ množica vseh permutacij množice N in naj bo $Pre^i(\mathcal{O})$ množica igralcev, ki so predniki igralca i v vrstnem redu $\mathcal{O} \in \pi(N)$. Potem lahko zapišemo aproksimacijo enačbe (3):

$$\varphi_i(\Delta) = \frac{1}{n! \cdot |\mathcal{A}_N|} \sum_{\mathcal{O} \in \pi(N)} \sum_{y \in \mathcal{A}_N} \left(f\left(\tau\left(x, y, Pre^i(\mathcal{O}) \cup \{i\}\right)\right) - f\left(\tau\left(x, y, Pre^i(\mathcal{O})\right)\right)\right) \tag{4}$$

S tem dobimo naslednji algoritem, ki s pomočjo vzorčenja dovolj dobro aproksimira rešitev.

Algoritem za računanje i-tega prispevka za primer $x \in \mathcal{A}_N$

---

določi število m, ki predstavlja željeno število vzorcev

$\varphi_i \longleftarrow 0$

for $i = 1$ to m **do**

izberi naključno permutacijo atributov $\mathcal{O} \in \pi(N)$

izberi naključni primer $y \in \mathcal{A}_N$

$v_1 \longleftarrow f\left(\tau\left(x, y, Pre^i(\mathcal{O}) \cup \{i\}\right)\right)$

$v_2 \longleftarrow f\left(\tau\left(x, y, Pre^i(\mathcal{O})\right)\right)$

$\varphi_i \longleftarrow \varphi_i + (v_1 - v_2)$

end for

$\varphi_i \longleftarrow \frac{\varphi_i}{m}$

Časovna zahtevnost in druge podrobnosti razlagalne metode so opisane v [1].

2.3 Razlaga modela

S pomočjo algoritma za razlago posameznih primerov lahko razlagamo tudi celotni model. Razlaga modela nam nudi celovit pregled nad tem, kako model dodeljuje vrednost regresijski spremenljivki glede na posamezen atribut in njegove vrednosti. Izgradnja razlage modela poteka tako, da vzamemo naključni primer, ki ima na željenem atributu določeno vrednost. Za primer z določeno vrednostjo sedaj izračunamo njegov prispevek - tako, kot bi to naredili za katerkoli nov primer (vključno z vzorčenjem). Kot prej omenjeno določeno vrednost izbiramo vrednosti iz celotnega spektera vrednosti atributa (seveda smiselno omejenim, če je interval prevelik - v naši nalogi je bilo izbrano število 100). Pri tem ostaja način računanja prispevkov nespremenjen. Za vsako različno vrednost atributa izračunamo prispevek in na koncu dobimo vektor števil, ki nam predstavlja spremembe prispevkov regresijske spremenljivke glede na različne vrednosti atributa. Ko to ponovimo za vsak atribut, dobimo skupek vektorjev števil, ki nam lepo predstavi delovanje modela. Primer vizualizacije take razlage kaže Slika 2.

3 EKSPERIMENTALNI DEL

3.1 Regresijski modeli

Regresijski modeli predstavljajo funkcije, ki slikajo prostor atributov v napovedano vrednost. Naloga učnih algoritmov je iz množice vzorcev z znano regresijsko spremenljivko izračunati funkcijo, ki jo uporabljamo za napovedovanje novih primerov. Regresijske modele ločimo po načinu učenja in obliki preslikovalne funkcije. Sledi kratek opis metod, uporabljenih v nalogi (podrobnejši opis teh in drugih metod strojnega učenja najdemo v [3]).

K-NN

Metoda *k* najbližjih sosedov temelji na lokalnosti. Izbere se *k* sosednjih primerov, ki so najbližje primeru, kateremu določamo vrednost. Nato se običajno izračuna povprečje teh vrednosti. Argument *k* je lahko poljubno število, vendar ne sme biti preveliko (premajhna ločljivost) in ne premajhno (prevelik vpliv šuma). Izbira velikosti argumenta *k* je odvisna od problema, ki ga rešujemo.

Linearna Regresija

Pri linearni regresiji se predpostavlja, da obstaja linearna relacija med vrednostmi atributov ter regresijsko spremenljivko. Koeficiente linearne funkcije, ki povezuje atribute z regresijsko spremenljviko, se določi tako, da se minimizira vsota kvadratov napak napovedi regresijske spremenljivke preko vseh učnih primerov. Rezultat je funkcija, ki preslika vrednosti atributov v regresijsko spremenljivko.

Regresijska Drevesa

Podobno kot pri odločitvenih drevesih izbirajo algoritmi za izgradnjo regresijskih dreves v vozliščih podmnožice vrednosti atributov. V vsakem vozlišču se nahaja atribut, v vejah pa so vrednosti. Vrednost novemu primeru običajno napovemo na podlagi znanih primerov, ki se nahajajo na istem listu.

Umetne nevronske mreže

Pri regresiji so najpogosteje uporabljene večnivojske umetne nevronske mreže. Sestavljene so iz več nivojev nevronov, ki so med seboj povezani. Vsaka povezava ima svojo utež, ki se jo določi pri procesu učenja nevronske mreže. Za regresijo je značilno, da se poleg vhodnih nevronov ter enega ali več skritih nivojev nevronov nahaja še izhodni nevron, ki ustreza regresijski spremenljivki.

SVM

Metoda temelji na implicitni transformaciji atributnega prostora s pomočjo jedrnih funkcij. Potrebno je optimizirati kriterijsko funkcijo, ki upošteva napake napovedi regresijske spremenljivke na učnih vzorcih ter kompleksnost funkcije.

3.2 Opis uporabljenih podatkovnih domen

Za testiranje je bilo uporabljenih več umetno generiranih podatkovnih domen. Domene vsebujejo atribute, poimenovane z A1 do A5, ter regresijsko spremenljivko R. Podan je tudi pričakovan uspeh algoritmov, uporabljenih v nalogi na posameznih domenah.

Cluster

Domena opisuje množico področij, ki jih določata atributa A1 ter A2, v katerih so naključno porazdeljene vrednosti iz določenih intervalov (interval določa območje). Vsi atributi so zvezni in zavzemajo vrednosti med 0 in 100. Pričakovati gre, da se na tej množici najbolje obneseta regresijsko drevo ter metoda najbližjih sosedov.

Pravila za generiranje vrednosti R:

$A1 \in [0{,}33] \vee A2 \in [0{,}33] \longrightarrow R = [0 - 10]$
$A1 \in [0{,}33] \vee A2 \in [33{,}66] \longrightarrow R = [70 - 80]$
$A1 \in [0{,}33] \vee A2 \in [66{,}100] \longrightarrow R = [30 - 40]$
$A1 \in [33{,}66] \vee A2 \in [0{,}33] \longrightarrow R = [90 - 100]$
$A1 \in [33{,}66] \vee A2 \in [33{,}66] \longrightarrow R = [50 - 60]$
$A1 \in [33{,}66] \vee A2 \in [66{,}100] \longrightarrow R = [20 - 30]$
$A1 \in [66{,}100] \vee A2 \in [0{,}33] \longrightarrow R = [10 - 30]$
$A1 \in [66{,}100] \vee A2 \in [33{,}66] \longrightarrow R = [70 - 100]$
$A1 \in [66{,}100] \vee A2 \in [66{,}100] \longrightarrow R = [40 - 50]$

Disjunct

Domena opisuje disjunkcijo med tremi atributi. Vsi atributi so zvezni in zavzemajo vrednosti med 0 in 100.

Pravila za generiranje vrednosti R:

if $(A1 > 50 \vee A2 > 40 \vee A3 > 60)$ $R = 1$; else $R = 0$

DisjunctBin

Tako kot domena Disjunct tudi ta domena opisuje disjunkcijo med temi atributi, vendar so tokrat atributi binarni.

Pravila za generiranje vrednosti R:
if $(A1 == 1 \vee A2 == 1 \vee A3 == 1)$ $R = 1$; else $R = 0$

Redundant

Preprosta linearna zveza med atributoma A1 ter A2, vendar so vrednosti nekaterih atributov identične (so torej redundantni). Na tej domeni testiramo, kako se različna obravnava redundantnih atributov pozna na razlagi napovedi. Kljub temu je osnova še vedno linearna funkcija, torej pričakujemo najboljše rezultate od linearne regresije.

Pravila za generiranje vrednosti R:
$R = 2 * A1 - 2 * A2$
$[A2 == A3], [A4 == A5]$

NonLinPoly

Običajna nelinearna kvadratna enačba z zveznimi atributi. Domena je bila izbrana zaradi težavnosti napovedovanja velikih števil, saj kvadrat atributov precej poveča zalogo vrednosti regresijske spremenljivke. Predpostavljamo dobro uvrstitev nevronske mreže ter regresijskih dreves.

Pravila za generiranje vrednosti R:
$R = 2 * A1^2 - 3 * A2^2 - A3$

NonLinSinCos

Nelinearna enačba, ki vsebuje kotne funkcije. Argumenti kotnih funkcij so izrazi, ki služijo normalizaciji. Domena je primer težjega problema, vendar zaradi omejene zaloge vrednosti regresijske spremenljivke predstavlja nekoliko lažjo domeno kot *NonLinPoly*.

Pravila za generiranje vrednosti R:

$$R = \sin\left(\left(\frac{A1 - 100}{100}\right) * 2 * \pi\right) + \cos\left(\left(\frac{A2 - 100}{100}\right) * 2 * \pi\right)$$

LocLinear

Lokalno linearna domena, ki je linearna v območjih, ki jih definirata diskretna atributa A3 in A4 (z zalogo vrednosti {0,1}). Preostali atributi so zvezni med 0 in 100. Domena je nekoliko zapletena varianta domene Linear.

Pravila za generiranje vrednosti R:

$R = 5 * A1 + A2;$ if $A3 = 0$ and $A4 = 0$

$R = A1 - 4 * A2;$ if $A3 = 0$ and $A4 = 1$

$R = 2 * A1 + 8 * A2;$ if $A3 = 1$ and $A4 = 0$

$R = -2 * A1 - 3 * A2;$ if $A3 = 1$ and $A4 = 1$

Linear

Preprosta linearna enačba z zveznimi atributi med 0 in 100. Pričakujemo dobre rezultate večine algoritmov. Domena je po zahtevnosti najbolj preprosta in s tem dober pokazatelj kako deluje metoda razlage na osnovnih problemih.

Pravila za generiranje vrednosti R:

$$R = A1 + 2 * A2 + 3 * A3$$

Random

Naključna domena v kateri na regresijsko spremenljivko ne vpliva noben atribut (je popolnoma naključna). Atributi so zvezni med 0 in 100. Vsak uspeh regresijskih algoritmov na tej domeni je zgolj slučajen. Pri tej domeni bi morali biti praviloma vsi prispevki zelo blizu 0.

Pravila za generiranje vrednosti R:
R = random

Xor

Domena opisuje zvezo xor med atributi A1, A2 in A3. Vsi atributi so zvezni med 0 in 100. Glavna lastnost domene je, da so atributi medsebojno odvisni (torej sama vrednost enega atributa nam ne pove veliko če ne upoštevamo še ostalih odvisnih atributov). Pričakovati gre, da se bo najbolje obnesla nevronska mreža. Linearna regresija ter KNN pa ne bi smeli biti preveč uspešni.

Pravila za generiranje vrednosti R:
$R = (A1 + A2 + A3)\ mod\ 5$

XorBin

Podobno kot pri domeni Xor, tudi tu opisujemo zvezo xor med atributi A1, A2 in A3, vendar so tokrat atributi binarni. Binarna varianta problema je nekoliko lažja za določene algoritme, vendar še vedno dovolj težka če ne znajo iskati odvisnosti med atributi. Tako kot pri *Xor* domeni pričakujemo tudi tu, da se bo najbolje obnesla nevronska mreža. Slabše rezultate lahko pričakujemo od linearne regresije ter KNN.

Pravila za generiranje vrednosti R:
$R = ((A1\ xor\ A2)\ xor\ A3)$

3.3 Metoda ocenjevanja uspešnosti razlage

Da bi lahko ocenili, ali je neka razlaga napovedi uspešna ali ne, moramo najprej definirati, kaj je uspešna razlaga. Ker za večino podatkov, nad katerimi računamo, poznamo točno vrednost regresijske spremenljivke za dan nabor vrednosti atributov (poznamo pravila za generiranje vrednosti R), lahko definiramo *razlago napovedi pravilnega modela*: v algoritmu (opisanem v poglavju 2.2) namesto napovedi napovedovalnega algoritma uporabimo napoved pravilnega modela za dane vrednosti atributov. Seveda razlago napovedi pravilnega modela lahko računamo le na umetnih problemih, saj pri realnih podatkih ne poznamo pravih vrednosti (oziroma pravila za izračun le-teh). Prispevek napovedi pravilnega modela nam tako določa prispevek, ki bi ga dobili, če bi imeli regresijsko funkcijo, ki nam vedno napove pravilen rezultat. Sedaj lahko izračunamo uspešnost oz. natančnost razlage kot evklidsko razdaljo dane razlage do razlage napovedi pravilnega modela. Vzemimo zopet $\mathcal{A}$ kot prostor atributov in z $predDiff_i$ označimo izračunano razlago s pomočjo regresijskih algoritmov na i-tem atributu za primer x. Podobno pa naj nam izraz $trueVal_i$ predstavlja razlago napovedi pravilnega modela.

$$eucl\_dist(x) = \sqrt{\sum_{i \in \mathcal{A}} (predDiff_i - trueVal_i)^2} \tag{5}$$

To razdaljo uporabimo za merjenje razdalje med dvema razlagama. V poglavju 3.5 jo uporabimo, kjer primerjamo napovedovalne algoritme s pomočjo povprečne evklidske razdalje. Pričakujemo, da bo pri boljših modelih tudi razlaga bližje razlagi napovedi pravilnega modela. In obratno, pri neuspešnih modelih pričakujemo razlago, ki bo oddaljena od razlage napovedi pravilnega modela.

3.4 Implementacija

Celotna naloga je bila implementirana v programskem jeziku Java [4]. Kot programerski vmesnik je bilo uporabljeno orodje Eclipse [5]. Napovedovalni algoritmi so bili uvoženi iz programskega paketa za strojno učenje Weka [6]. Uporabljene so bile še knjižnjica za statistično obdelavo JSC [8] ter knjižnica za grafični zapis v datoteke *.eps* [7]. Za izračune je bil uporabljen namizni računalnik (3GHz CPU (dual core), 3GB RAM).

Za računanje regresijske spremenljivke smo uporabili različne algoritme, da bi dobili tembolj barvit spekter glede na način izračuna. Uporabili smo sledeče: linearna regresija (LR), regresijsko drevo (M5P), nevronska mreža (MP), metoda podpornih vektorjev (SVM) ter dve verziji *k* najbližjih sosedov za k=1 in k=10 (KNN1 in KNN10).

Učne množice so bile praviloma velike 1000 primerov, testne množice prav tako 1000 primerov. Izjemi sta algoritma KNN1 in KNN10, katera je bilo potrebno omejiti na 500 primerov učne množice zaradi počasnega napovedovanja regresijske spremenljivke. Število vzorcev je bilo 10000, izjema sta oba KNN algoritma. Za slednja smo pri vzorčenju izbrali 5000 vzorcev. V nalogi se nismo veliko posvečali časovni zahtevnosti. Hoteli smo le zagotoviti razlago posameznega primera v doglednem času (torej ne več kot nekaj sekund). Generiranje učnih množic (opisanih v 3.2) je bila spisana kot pomožni program. Za shranjevanje smo uporabili metode iz Weke, ki shranijo podatkovne množice v formatu ARFF [9].

Za osnovo je služil program, ki sta ga razvila in opisala Štrumbelj in Kononenko [1]. Potrebna je bila prilagoditev na regresijo in regresijske algoritme, napisati generator podatkov ter procedure za računanje evklidske razdalje, razlago modelov ter prilagoditi vizualizacijo za regresijo. Vizualizacija razlage modelov ter evklidske razdalje je bila realizirana na podoben način kot vizualizacija razlage napovedi.

3.5 Analiza rezultatov

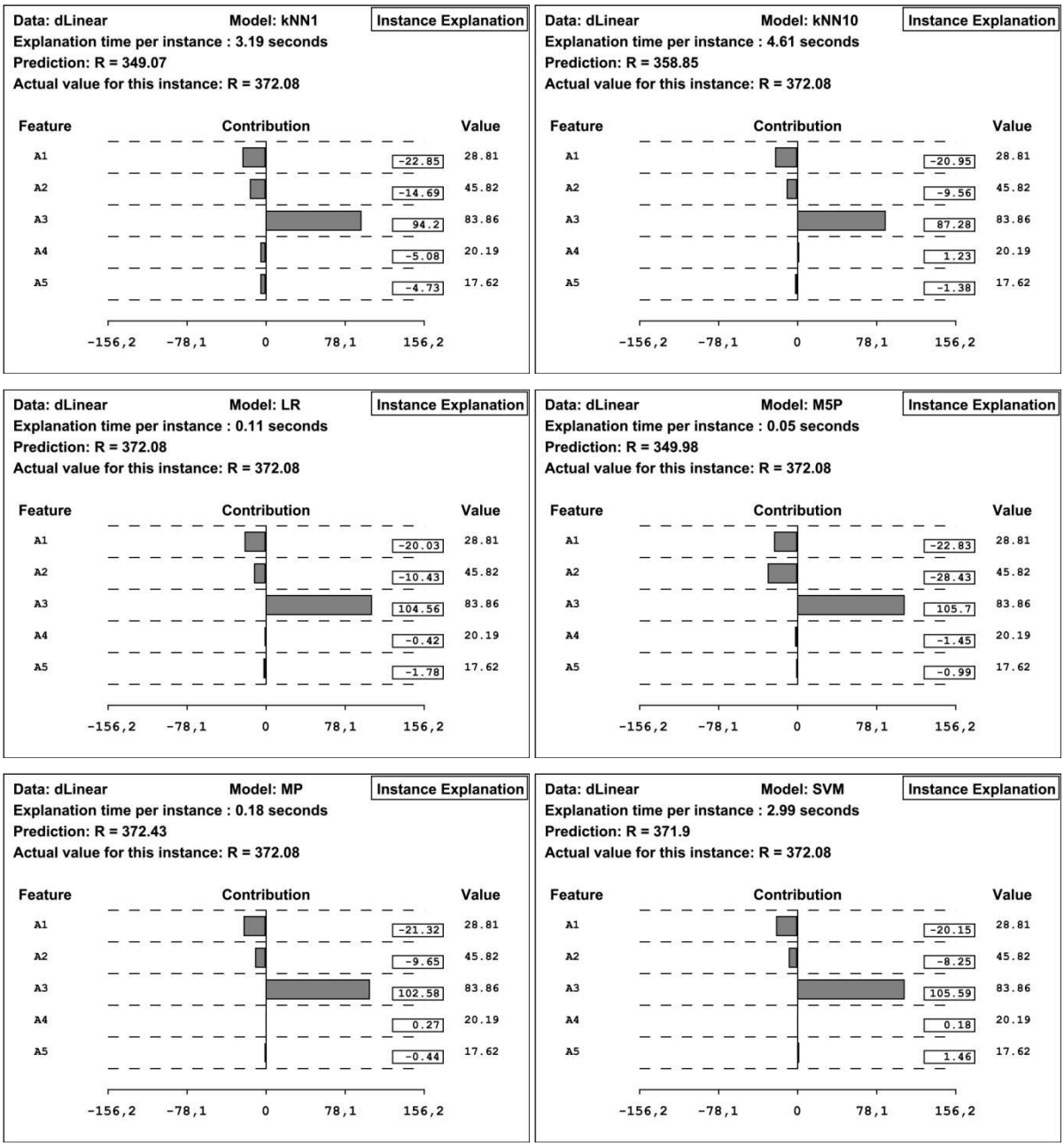
Našo metodo razlage smo preizkusili na umetnih domenah (opisanih v 3.2). Iz rezultatov želimo ugotoviti splošno uporabnost metode ter morebitne razlike med algoritmi oz. odstopanja od pričakovanih vrednosti. Kot kažejo rezultati (glej Tabelo 1), nam je to v veliki meri tudi uspelo. V kategoriji uspešnosti napovedovalnosti algoritmov je zaostajal SVM predvsem zaradi splošnih nastavitev, saj bi bilo potrebno konfigurirati SVM za vsak specifičen problem posebej, da bi dobili boljše rezultate. Nekoliko slabše sta jo odnesla tudi oba algoritma KNN, vendar je glavni razlog za to 50% manjša učna množica ter tudi 50% manj vzorcev pri vzorčenju. Razlog za manjšo učno množico ter manj vzorcev pa tiči v počasnem napovedovanju algoritma. Tu se skriva tudi največja pomanjkljivost pristopa. Napovedovalnim algoritmom ni potrebno graditi modela za vsak vzorec, vendar pa je treba večkrat napovedati vrednosti za posamezen vzorec. Če je hitrost te napovedi počasna, se ta počasnost multiplikativno prenese na počasnost razlagalnega algoritma. V primeru KNN je tipično, da je hitrost gradnje algoritma zelo hitra in napovedovanje tem počasnejše, čim večjo

množico učnih primerov imamo. Tudi pri SVM izbira jedra vpliva na hitrost napovedovanja novih primerov (v nalogi smo uporabili nekoliko počasnejše jedro, ki pa je splošno uspešnost algoritma malenkostno povečala).

	LR	MP	SVM	M5P	kNN1	kNN10
NonLinSinCos	0,78 0,84 83,37	0,20 0,34 33,30	0,79 0,85 83,38	0,13 0,18 17,75	0,35 0,57 59,35	0,43 0,49 51,59
DisjunctBin	0,16 0,24 79,01	0,01 0,00 0,00	0,23 0,32 104,74	0,01 0,00 0,00	0,02 0,00 0,00	0,02 0,00 0,00
Disjunct	0,17 0,27 85,63	0,13 0,23 72,18	0,23 0,34 105,98	0,03 0,11 33,63	0,10 0,23 75,13	0,12 0,19 60,53
Cluster	19,99 28,14 99,06	20,57 29,60 104,21	19,55 28,42 103,31	2,09 7,63 26,86	12,88 27,99 98,51	14,72 22,51 79,19
Redundant	2,33 0,00 0,00	37,74 1,41 1,74	35,98 0,33 0,41	7,20 9,38 11,66	37,12 13,79 16,35	37,39 10,58 13,10
NonLinPoly	9427,60 10090,32 100,10	1991,20 3208,44 31,83	9420,52 10111,63 101,20	1653,79 2399,02 23,80	4029,83 7309,68 72,53	4690,03 5889,41 58,44
LocLinear	112,72 170,11 49,57	13,75 18,50 5,17	111,71 179,14 52,90	20,38 27,63 7,72	25,53 42,80 11,93	28,73 39,52 11,02
XorBin	0,29 0,50 100,44	0,20 0,39 78,71	0,37 0,57 142,85	0,08 0,31 62,50	0,09 0,32 63,31	0,24 0,41 85,65
Xor	0,72 1,44 100,00	0,74 1,59 110,61	0,73 1,44 100,38	0,75 1,46 101,39	1,03 1,98 140,61	0,78 1,51 104,01
Random	1,82 29,07 100,46	1,72 41,20 142,39	3,19 29,48 101,88	4,01 30,11 104,06	14,82 41,28 138,64	5,79 30,21 104,43
Linear	3,09 0,00 0,00	3,10 0,31 0,29	3,08 0,41 0,37	18,38 26,24 24,12	19,80 35,15 33,57	22,12 27,10 24,02

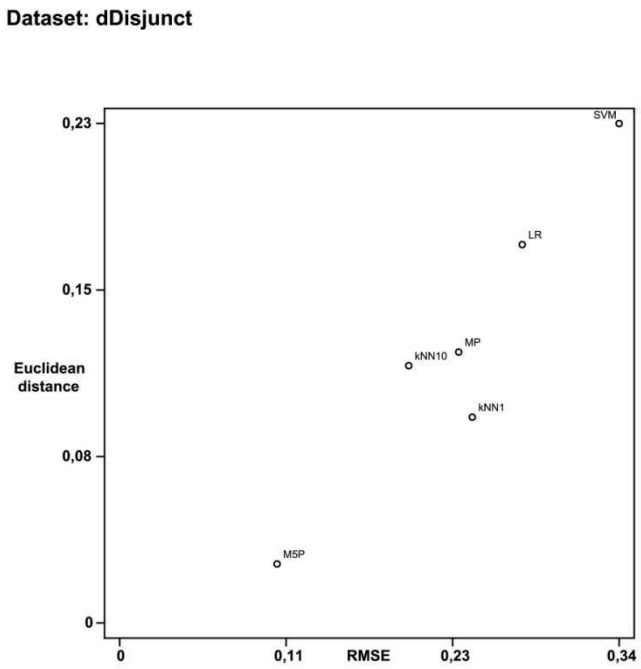
Tabela 1: Uspešnosti razlag ter uspešnosti napovedovalnih algoritmov na posamezi podatkovni domeni. Zgornje število v celici predstavlja evklidsko razdaljo izračunane razlage do razlage napovedi pravilnega modela, srednje število predstavlja RMSE algoritma, spodnje število predstavlja RRMSE (izraženo v odstotkih).

Če želimo imeti univerzalno metodo za razlago napovedi, potem se morajo razlage različnih algoritmov z dovolj podobno uspešnostjo razlage za isti primer ujemati. Kot vidimo na sliki 3 sicer ujemanje ni popolno, vendar je treba upoštevati tudi dejstvo, da se natančnost algoritmov še vedno nekoliko razlikuje (na sliki 7 so te razlike prikazane). Lepo pa je vidno, da je večinski del napovedi enak ali zelo podoben za vse algoritme.



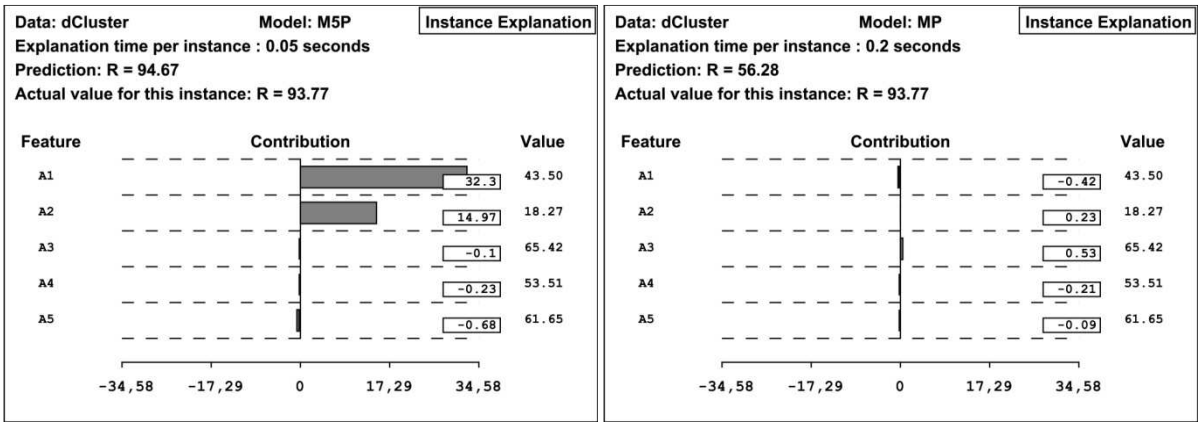
Slika 3: Primerjava razlag za isti primer iz domene Linear z različnimi napovedovalnimi algoritmi.

Ena izmed željenih lastnosti naše razlagalne metode je med drugim tudi ta, da se s slabšanjem natančnosti napovedi slabša tudi kvaliteta razlage napovedi. Velja tudi obratno, z boljšo natančnostjo napovedovalnega algoritma želimo boljše razlage. Da bi ugotovili, ali to velja za našo metodo, smo sestavili graf, ki kaže evklidsko razdaljo (opisana v 3.3) do razlage napovedi pravilnega modela posameznih napovedovalnih algoritmov v odvisnosti od natančnosti algoritma (izražen z RMSE [3]). Željen potek grafa je linearen (primer kaže slika 4).



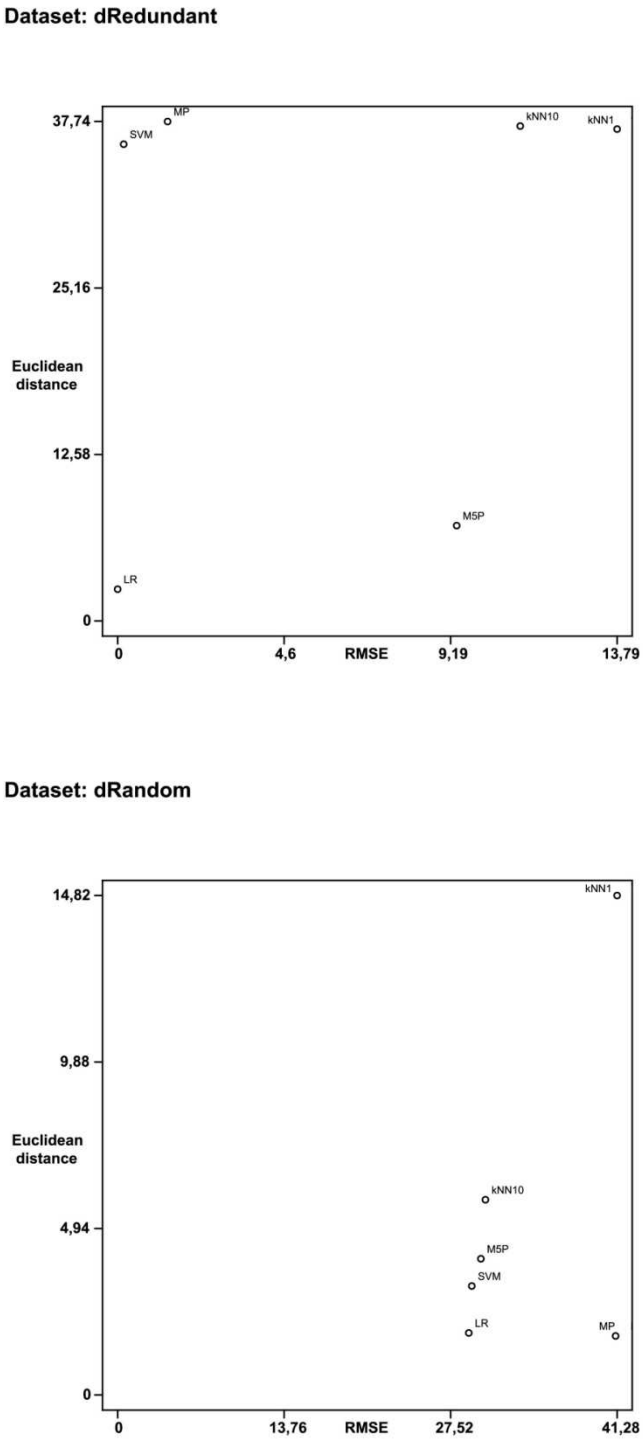
Slika 4: Natančnost razlage v odvisnosti od natančnosti napovedi posameznih napovedovalnih algoritmov za domeno Disjunct. Linearna ureditev algoritmov nam kaže uspešnost razlagalne metode.

Primer različnih razlag nam kaže slika 5. Kot vidimo, lahko za isti primer dobimo dve različni razlagi. Razlog za veliko razliko pa je razlika v natančnosti napovedi algoritmov. Model M5P se je najbolje odrezal na tej podatkovni domeni, MP pa se je slabše obnesel (uspešnost algoritmov prikazana v obliki grafa na sliki 7).

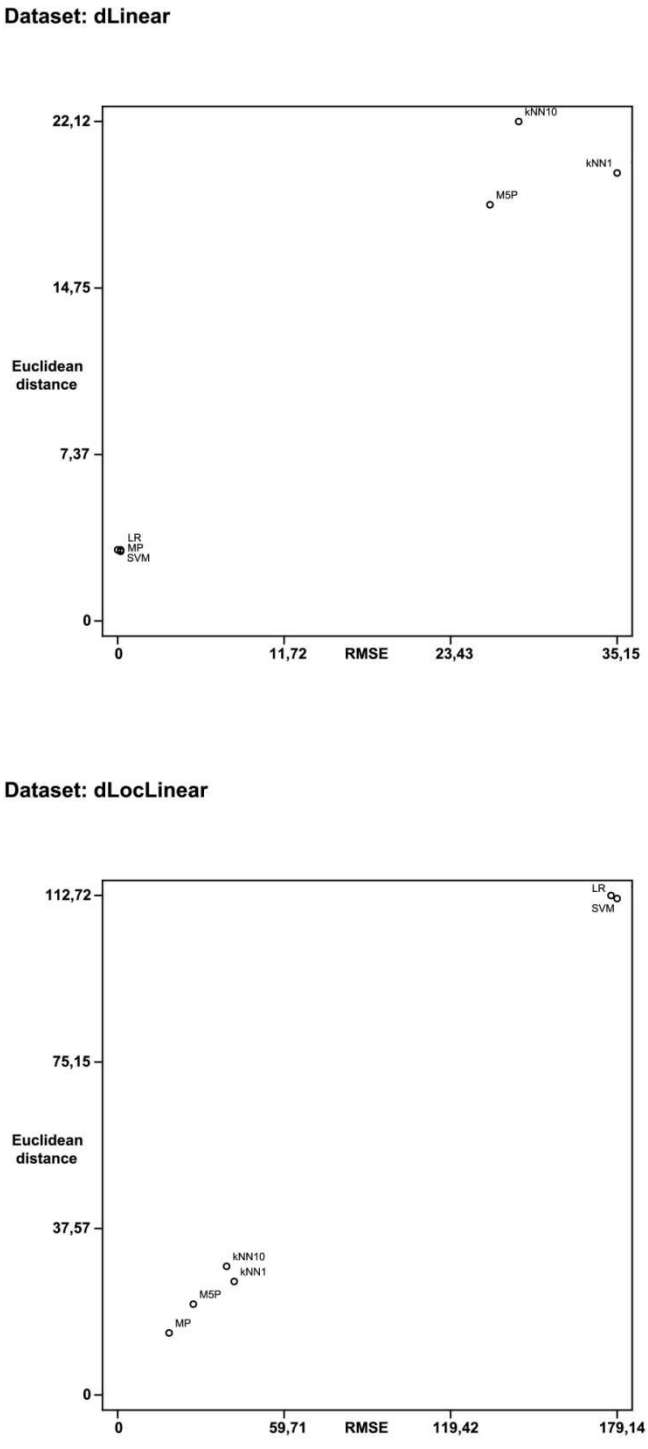


Slika 5: Primer razlage istega primera iz domene Cluster dveh različnih napovedovalnih algoritmov. Različna natančnost algoritmov daje različne razlage.

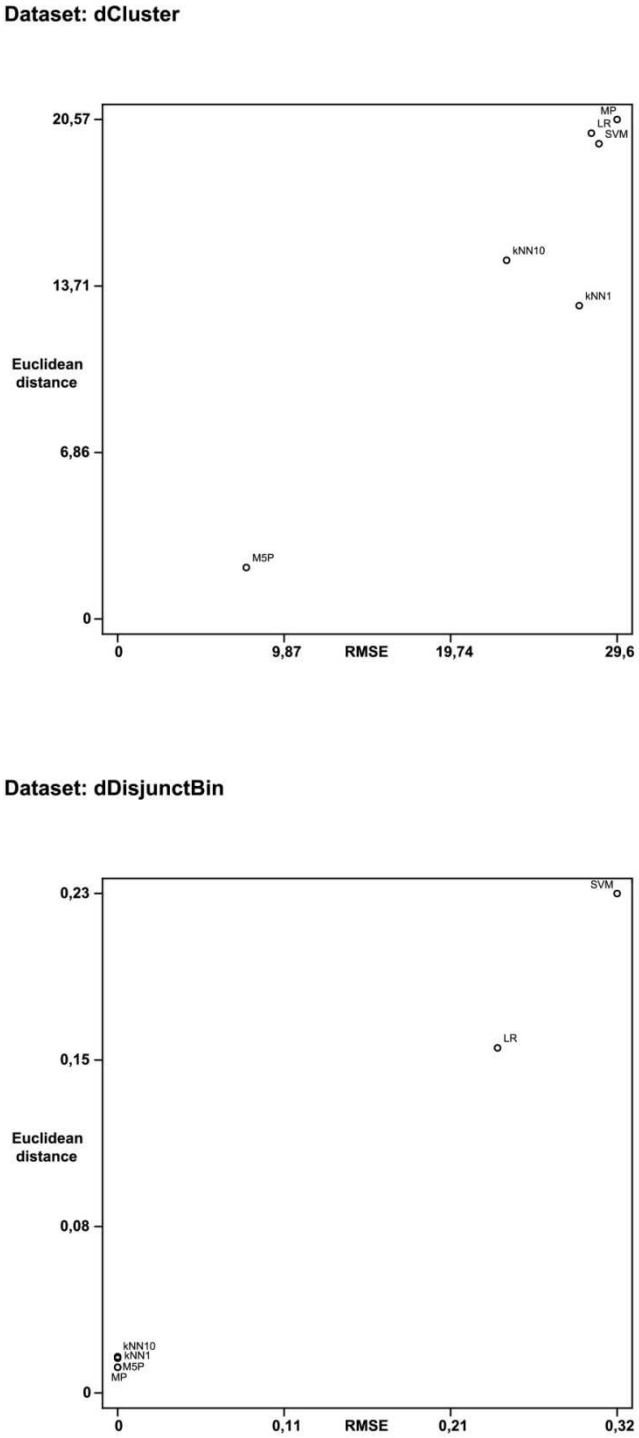
Skoraj vsak graf nad posamezno domeno vsebuje zgoraj navedeno linearno zvezo (glej Slika 4, 7 do 10). Izjemi sta le dva grafa: graf za podatkovno domeno *Random* ter graf za domeno *Redundant* (prikazana na Sliki 6). Kot prvo si poglejmo domeno *Random*. Na tem grafu se večina algoritmov nahaja v spodnjem desnem kotu grafa, torej s slabimi RMSE vrednostmi ter slabimi napakami razlage. Razlog za to so zelo nizke vrednosti prispevkov in posledično je tudi evklidska razdalja do vrednosti pravilnega modela majhna. Tu izstopa le algoritem KNN1, ki zaradi prevelikega prilagajanja učni množici obtiči v zgornjem delu grafa z veliko evklidsko razdaljo. Nekoliko drugačen problem je pri domeni *Redundant*. Kot vidimo na sliki 6 izstopata dva algoritma, ki tičita v zgornjem levem kotu. To sta algoritma SVM ter MP. Tudi oba KNN algoritma sta precej visoko na evklidski razdalji. Razlog za težavo tiči v domeni sami ter načinu izračuna vrednosti prispevkov pravilnega modela redundantnih atributov. Za izračun vrednosti pravilnega modela namreč uporablja generatorska funkcija, ki uporablja atribut A2. Če bi hoteli, da uporablja oba atributa, bi morali v generacijsko funkcijo vdelati 50% možnost da uporabi ali A2 ali A3. Tako so izračunani prispevki pravilnega modela za atribut A2 zelo visoki in A3 ima zelo nizke prispevke. Zgolj slučajno sta tudi linearna regresija ter algoritem regresijskih dreves zanemarila atribut A3 (saj je isti kot A2) in tako dobila nizke evklidske razdalje do prispevkov pravilnega modela. Ostali algoritmi pa so prispevke za A2 ter A3 enakomerno razporedili po obeh atributih. Zaradi te razlike je izračunana evklidska razdalja veliko večja. Vsekakor pa napaka tu ni v metodi razlage, ampak v definiciji razlage napovedi pravilnega modela. V primeru redundantnih atributov imamo pač več različnih možnih razlag, kar nam pokaže neko pomanjkljivost v merjenju razdalje od razlage napovedi pravilnega modela.



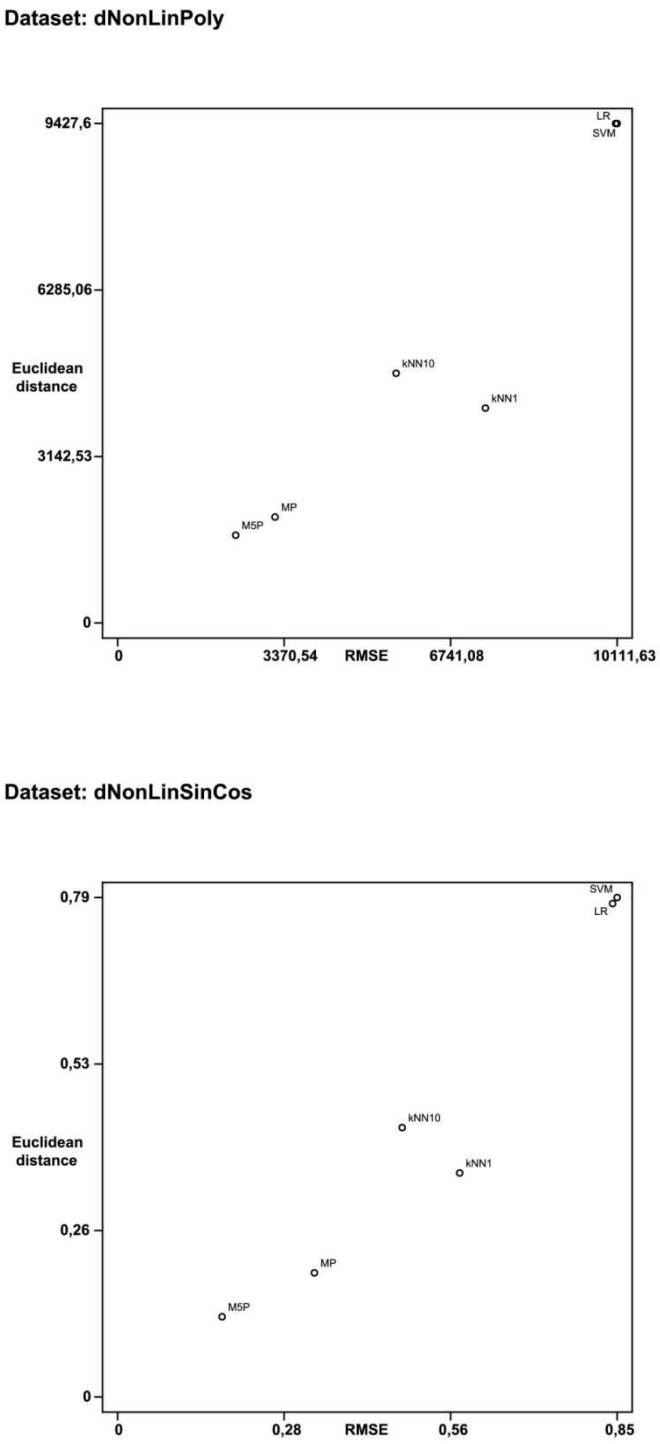
Slika 6: Natančnost razlage v odvisnosti od natančnosti napovedi za domeno Redundant (zograj) in Random (spodaj). V primeru Redundant nelinearna urejenost algoritmov kaže na problem z obravnavo redundantnih atributov.



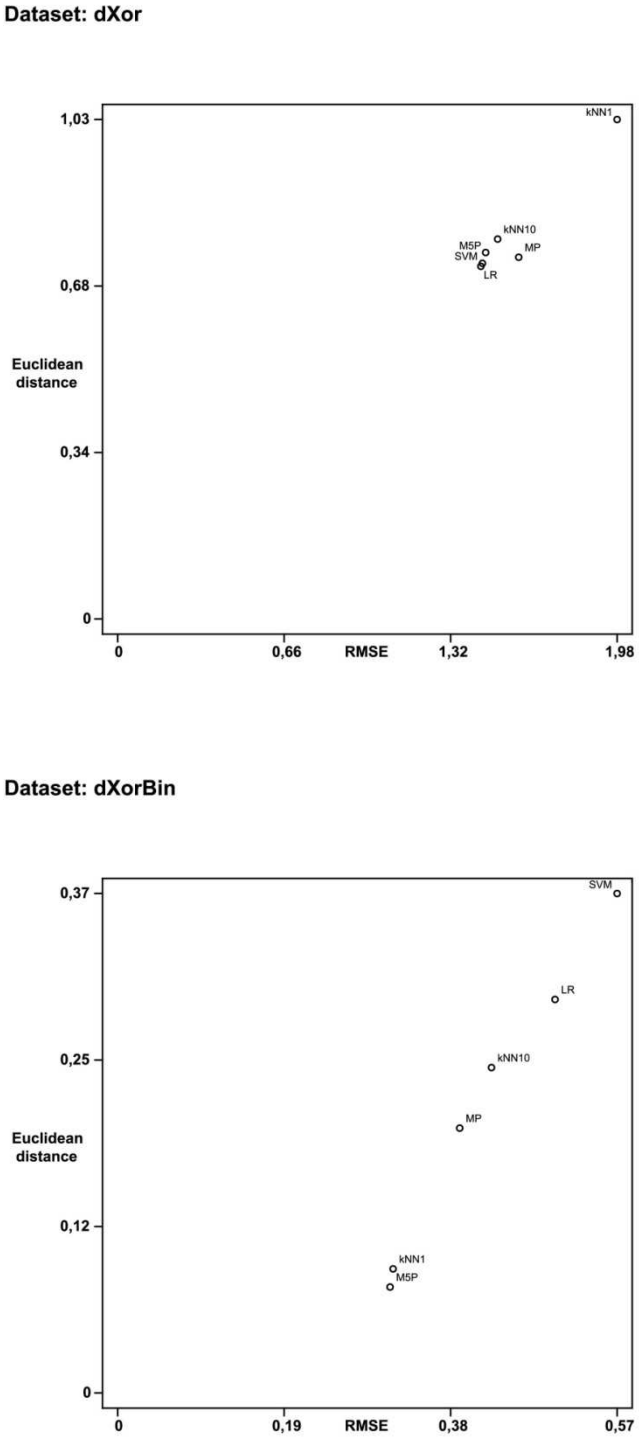
Slika 7: Natančnost razlage v odvisnosti od natančnosti napovedi za domeno Linear (zgoraj) in LocLinear (spodaj).



Slika 8: Natančnost razlage v odvisnosti od natančnosti napovedi za domeno Cluster (zgoraj) in DisjunctBin (spodaj).



Slika 9: Natančnost razlage v odvisnosti od natančnosti napovedi za domeno NonLinPoly (zgoraj) in NonLinSinCos (spodaj).



Slika 10: Natančnost razlage v odvisnosti od natančnosti napovedi za domeno Xor (zgoraj) in XorBin (spodaj).

V prejšnjih poglavjih smo že omenili in opisali razlago modelov, ki nam pomagajo razumeti koncept delovanja napovedovalnega algoritma. Še en lep primer take razlage lahko vidimo na Sliki 11. Domena združuje funkciji sinus in cosinus, ki ju lahko prepoznamo iz vizualizacije KNN na prvih dveh atributih. Prispevki ostali atributov so po velikosti dovolj majhni, da jim ne pripišemo velikega vpliva na postopek napovedovanja. Algoritem LR pa ima težave s krivuljo, vendar relativno dobro aproximira smer sinusa na atributu A1.

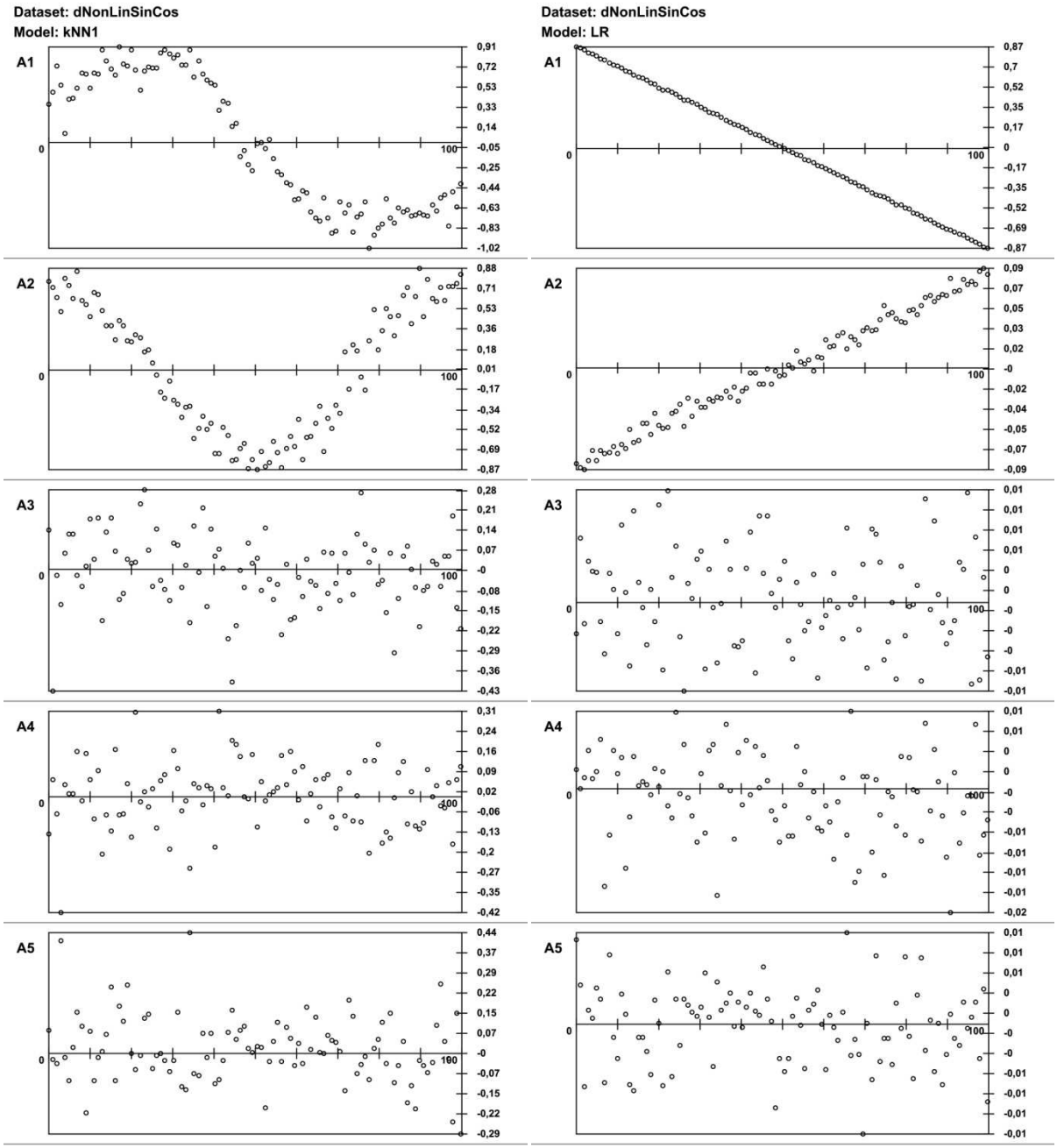
Podobno lahko opazimo stopničasto porazdelitev prispevkov regresijskih dreves na Sliki 13 (problem opisan v članku [2]). Lepo je viden način, kako regresijska drevesa delujejo in po koliko nivojev je bilo uporabljenih pri posameznem atributu. Vidna je tudi porazdelitev, za kakšne vrednosti atributa se napovejo določene vrednosti.

Vizualizacija razlage modelov nam lahko pomaga tudi pri iskanju neskladij. Vzemimo primer različnega načina računanja redundantnih atributov (Slika 12). Brez težav vidimo, da algoritem LR atributu A3 ne pripisuje velikega vpliva, algoritem MP pa mu pripisuje enak vpliv kot atributu A2. S pomočjo vizualizacije lažje vidimo vzrok za napako.

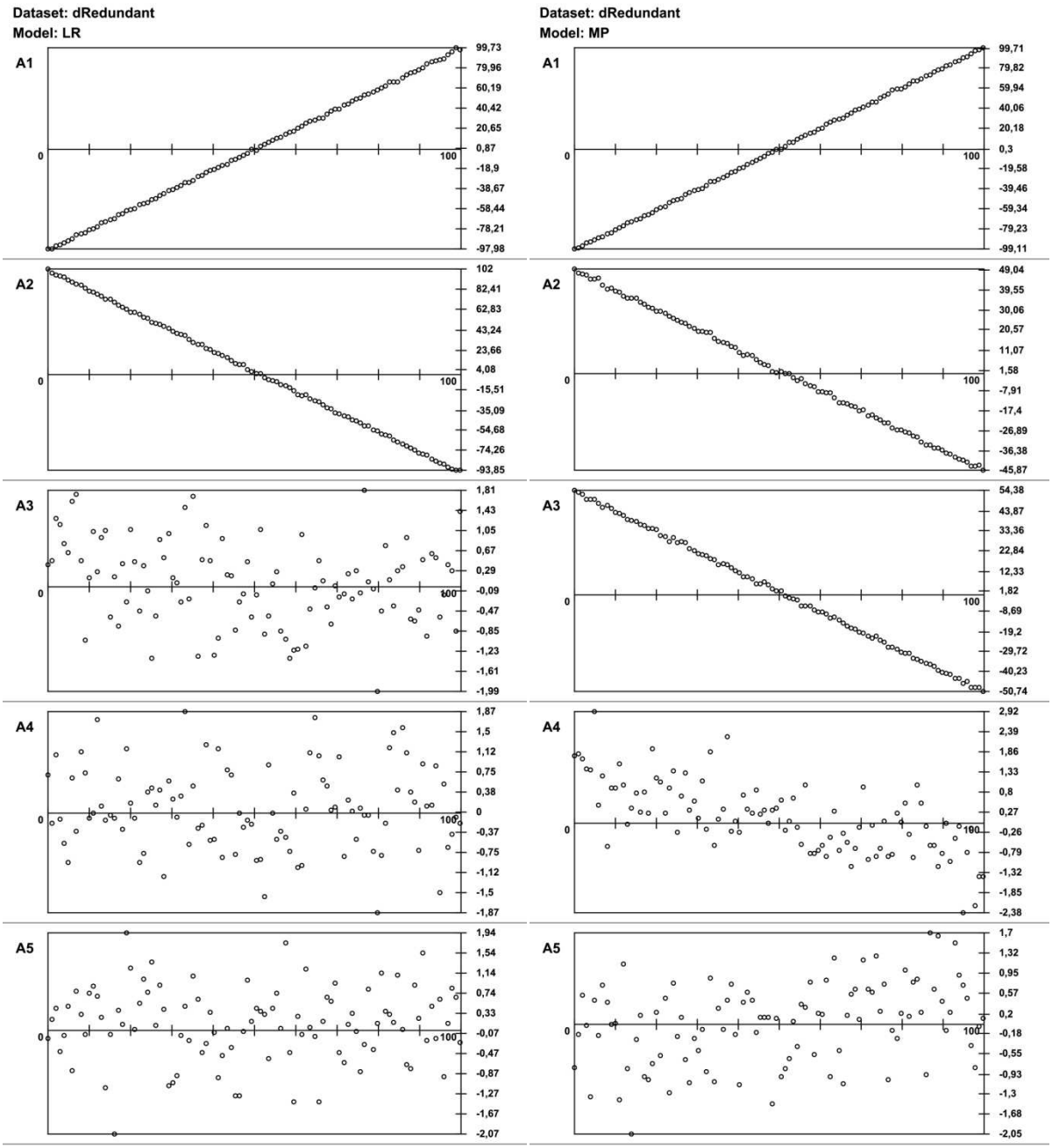
Na primeru, ki ga kaže Slika 14, vidimo, da četudi ima domena *Cluster* kompleksno zgradbo, še vedno lahko izluščimo osnovni trend atributa A2, ki se v začetnem in končnem delu spušča nižje, v osrednjem delu pa so prispevki višji. Ob natančnejšem pogledu pravil za generiranje spremenljivke R (glej poglavje 3.2) vidimo, da se po večini trend na grafu ujema z dejanskimi števili.

Naslednji primer kaže kako nam lahko vizualizacija razlage modelov pomaga pri iskanju nepravilnosti. Slabo natančnost napovedi algoritmov KNN in M5P na preprosti domeni *Linear* (glej Tabela 1) lažje razumemo, ko vidimo njuni vizualizaciji na Sliki 15 (izbran predstavnik družine KNN je bil KNN10, vendar podobne ugotovitve veljajo tudi za KNN1). Po primerjavi z algoritmom MP na tej domeni (predstavljen na Sliki 2), ugotovimo, da algoritem KNN10 nenatančno določi linearno zvezo. Opazimo, da ima drugače linearen graf oba konca rahlo ukrivljena (podcenjujemo absolutno vrednost robnih primerov). Namreč, bolj ko gremo proti robu, več sosedov imamo na notranji strani, zato ti pri povprečenju blažijo naraščanje funkcije. Do ekstremnega primera naletimo, ko vzamemo skrajno levi (desni) primer, ki ima vse svoje sosede na desni (levi). Opazni so tudi zelo veliki prispevki nepomembnih atributov A4 in A5. V primeru M5P pa vidimo (četudi relativno dobro določi največje in najmanjše prispevke) slabo stran regresijskih dreves, in sicer nivojsko napovedovanje vrednosti. Podobno kot na domeni *Linear* opazimo podobno obnašanje prej omenjenih algoritmov na domeni *Redundant*. Vendar je osnova tudi tu linearna funkcija in za vse tri algoritme velja zgornja obrazložitev.

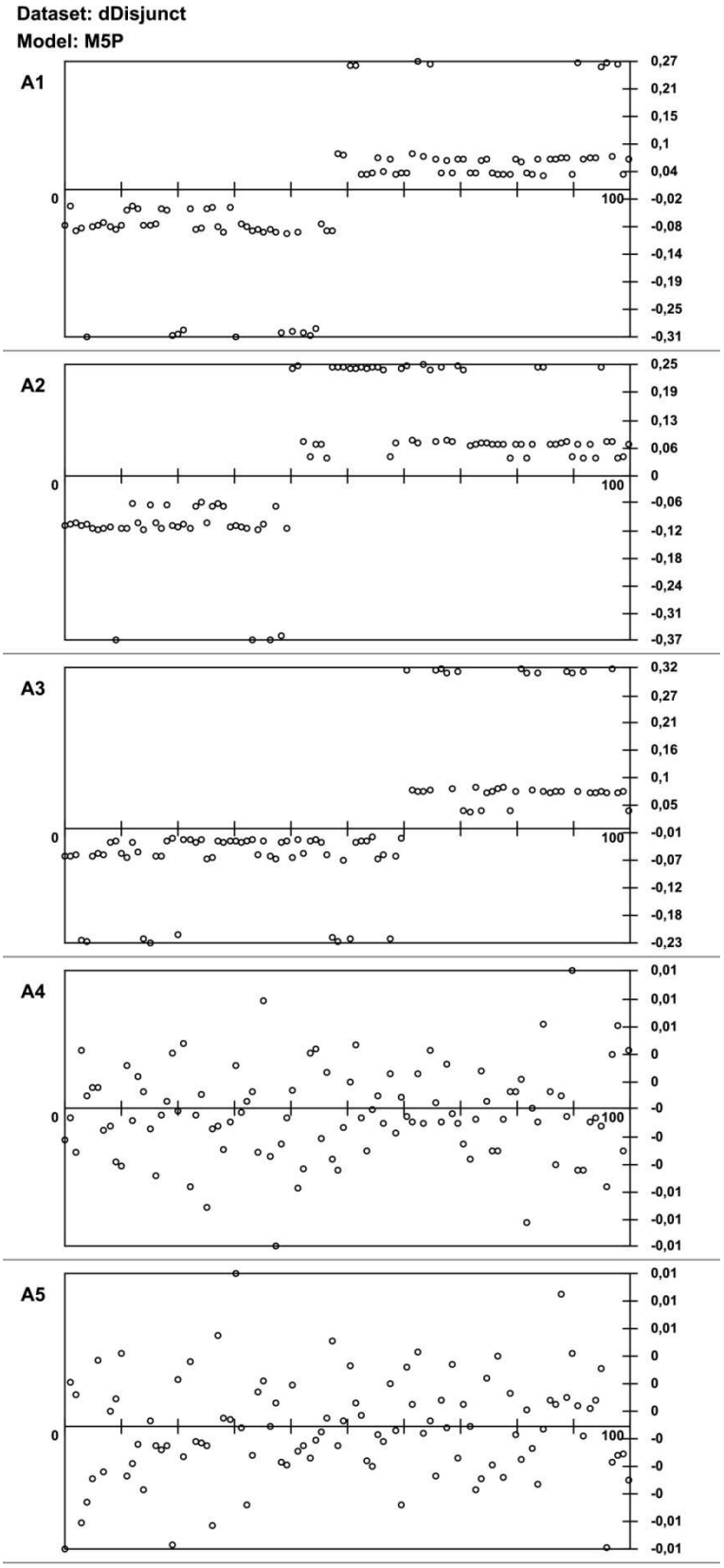
Vendar pa vizualizacija razlage modelov ne daje vedno prave slike. Pirmer za to kaže Slika 16, kjer vidimo, da je večina razlag posameznih atributov zelo blizu 0. Razlog za tako neupoštevanje prispevkov je visoka soodvisnost atributov na domeni *Xor*. Ker pri vizualizaciji razlage modelov ne upoštevamo soodvisnosti atributov, nam v teh primerih razlaga modelov ne pomaga.



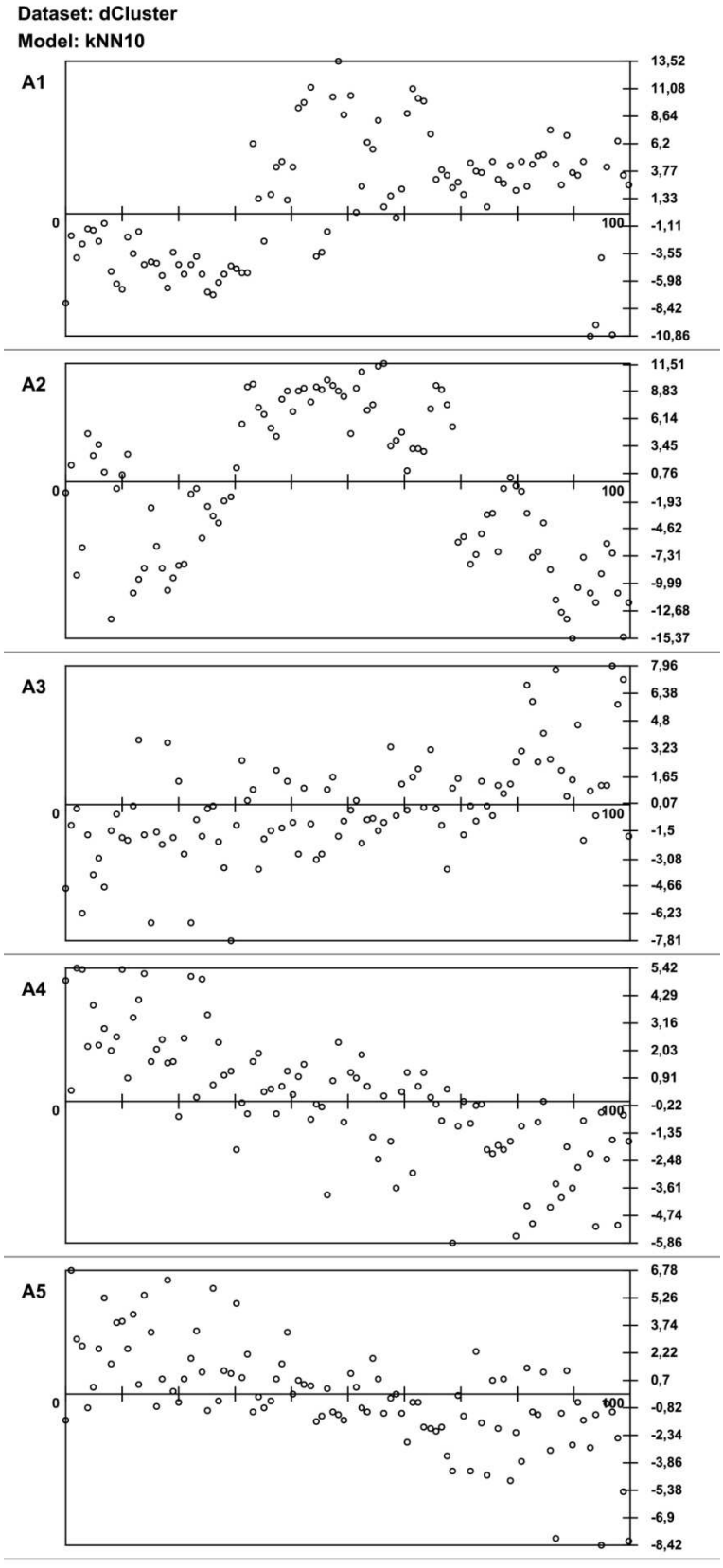
Slika 11: Razlaga modela KNN1(levo) in LR(desno) na podatkovni domeni NonLinSinCos. Kot vidimo ima LR težave s krivuljami.



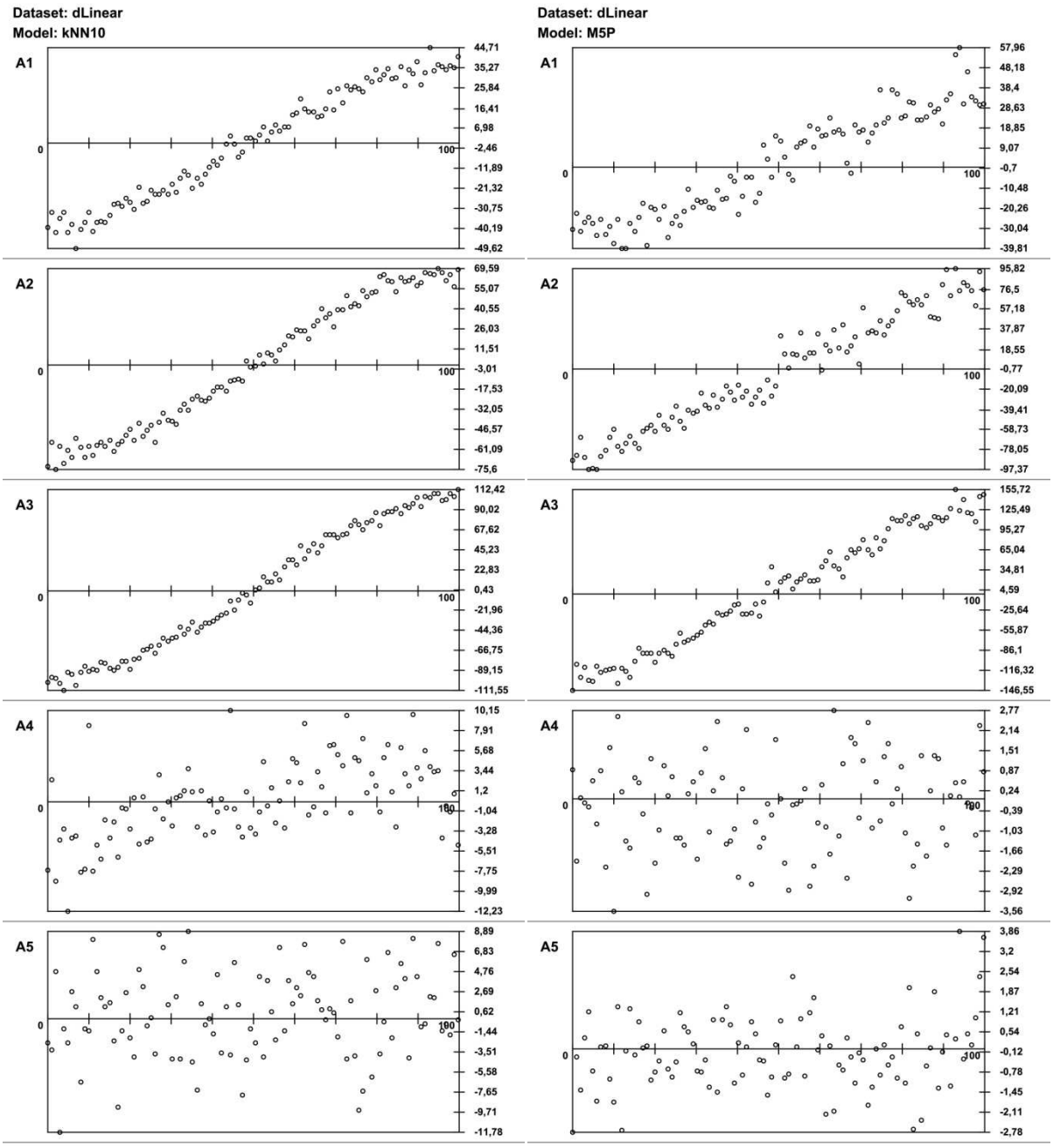
Slika 12: Razlaga modelov LR in MP na podatkovni domeni Redundant. Opazna razlika na atributu A3.



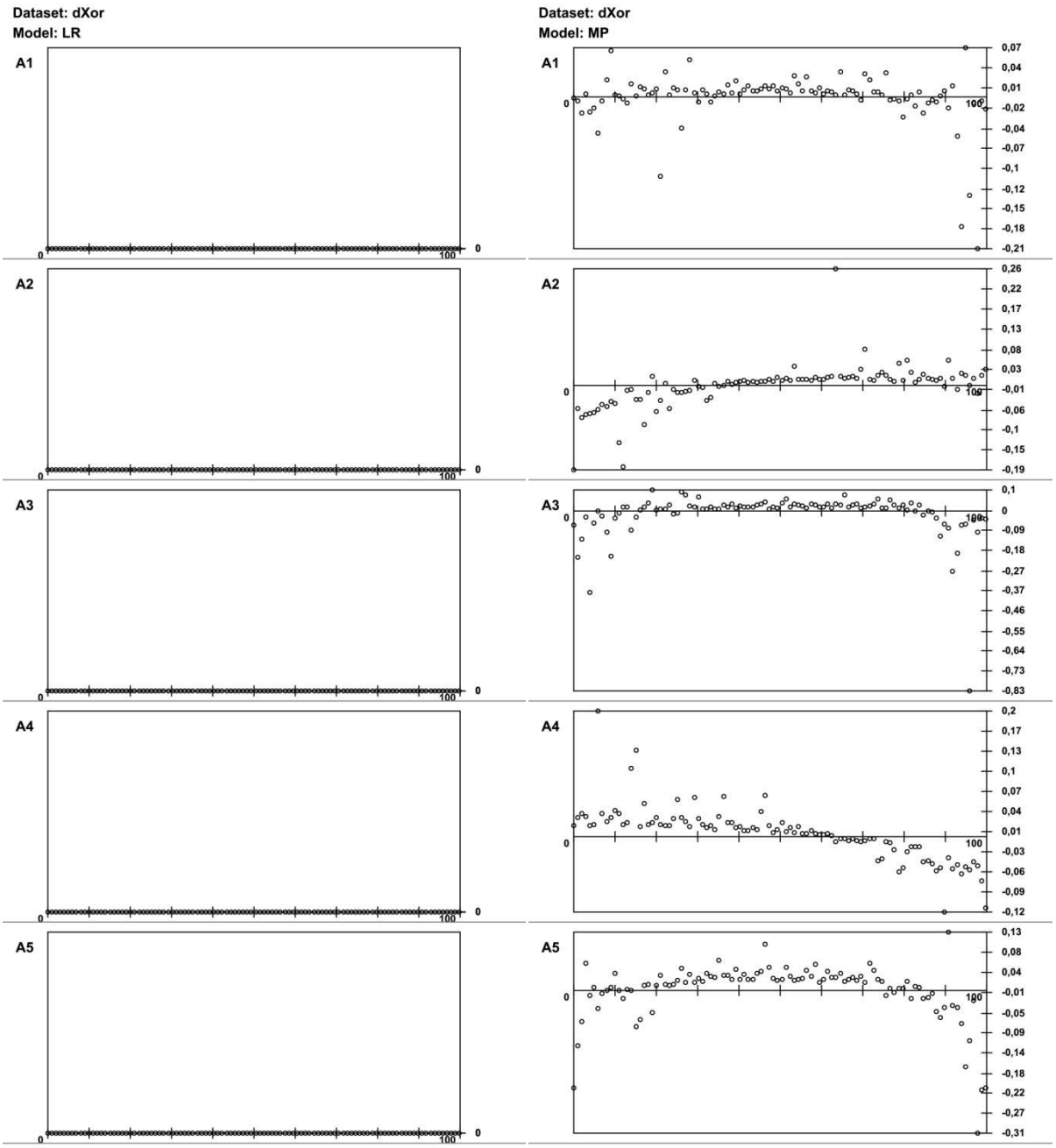
Slika 13: Razlaga modela M5P na podatkovni domeni Disjunct. Vidna je stopničasta razdelitev atributov A1, A2 in A3 (ostala atributa ne prispevata bistveno k napovedi).



Slika 14: Razlaga modela KNN10 na podatkovni domeni Cluster. Lepo so vidne mejne razmejitve med intervali domene (še posbej atributa A2) ter tipični prispevek s teh intervalov.



Slika 15: Razlaga modelov KNN10 in M5P na podatkovni domeni *Linear*. Razloge za slabe napovedi lahko razberemo iz vizualizacije razlage modelov.



Slika 16: Razlaga modelov LR in MP na podatkovni domeni *Xor*. Visoka soodvisnost atributov onemogoči dobro razlago modela.

3.6 Razlaga v kontekstu

V uvodu smo omenili, da naša metoda med drugim odpravlja problem lokalnosti, ki smo ga srečali pri metodi iz članka [2]. V tem poglavju bomo pokazali, kako lahko isti problem, izražen na različne načine, vpliva na razlago.

Zakaj si sploh želimo razlago v kontekstu? Pri razlagi izven konteksta vidimo, kako atributi vplivajo na model. Iz tega pa ne sledi, da zagotovo izvemo veliko o sami podatkovni množici. Izven konteksta pomeni, da smo predpostavili, da so vse kombinacije vrednosti atributov enako verjetne. Včasih se nam zato zgodi, da neupravičeno pripišemo velik prispevek neki vrednosti. Ta vrednost je sicer zelo pomembna v delovanju modela, čeprav bi ji uporabnik, zaradi njene pogostosti/redkosti, pripisal manjši prispevek, in obratno. Če želimo pri neenakomerno porazdeljeni podatkovni množici upoštevati kontekst, naletimo na dve vrsti težav:

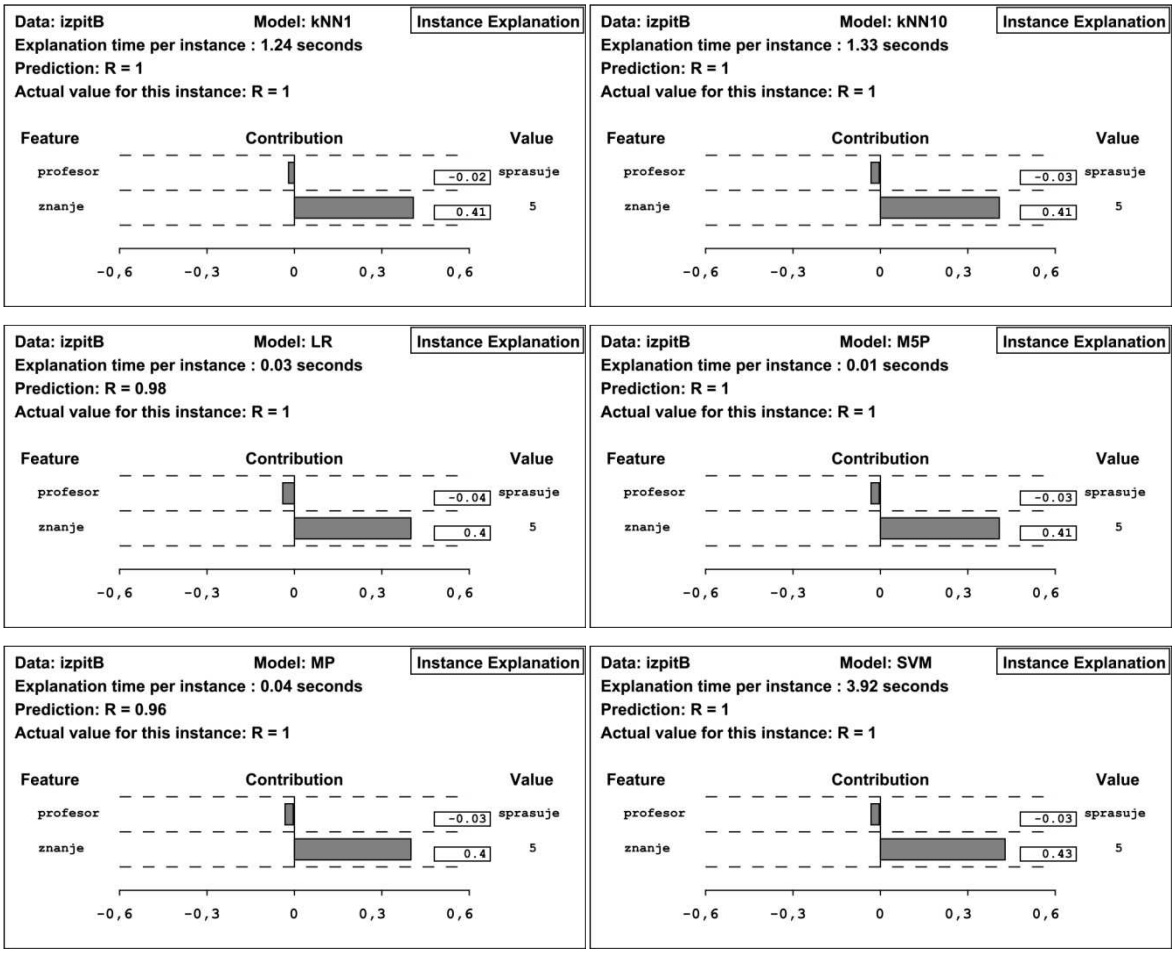
- Pri vzorčenju bi morali posamezne kombinacije vzorčiti v skladu z njihovo porazdelitvijo. To bi precej zapletlo postopek vzorčenja in povečalo časovno zahtevnost.
- V praksi težko aproksimiramo pravo porazdelitev, saj imamo ponavadi premalo podatkov.

Zato uporabimo preprosto rešitev. Vrednosti zanemarjenih atributov ($\notin S$), ki jih v osnovnem algoritmu izberemo naključno (y), sedaj prepišemo iz naključno izbranega primera iz učne množice. S tem implicitno predpostavimo, da so porazdelitve atributov medseboj neodvisne. Po predpostavki je torej verjetnost, da bo neka vrednosti izbrana, neodvisna od vrednosti atributov, ki jih ne zanemarimo.

Pri ilustraciji razlage v kontekstu bomo kot podatkovno domeno vzeli študente na ustnem izpitu. Profesor na izpitu bodisi vpraša študenta bodisi ne, kar izrazimo z binarnim atributom *profesor*. Če ga ne vpraša, potem študent izpit naredi. Če spraševanje, potem je uspeh na izpitu odvisen od znanja študenta, ki je izraženo z numeričnim atributom *znanje* (verjetnost, da študent naredi izpit: $P(opravi) = \frac{znanje}{5}$). Narejen izpit označimo z 1, neuspeh pa z 0. Regresijska spremenljvika je torej določena z enotno formulo in jo lahko interpretiramo kot verjetnost, da je študent izpit opravil. Preizkusili smo 5 različic tega problema. Edina razlika med njimi je v distribuciji atributov.

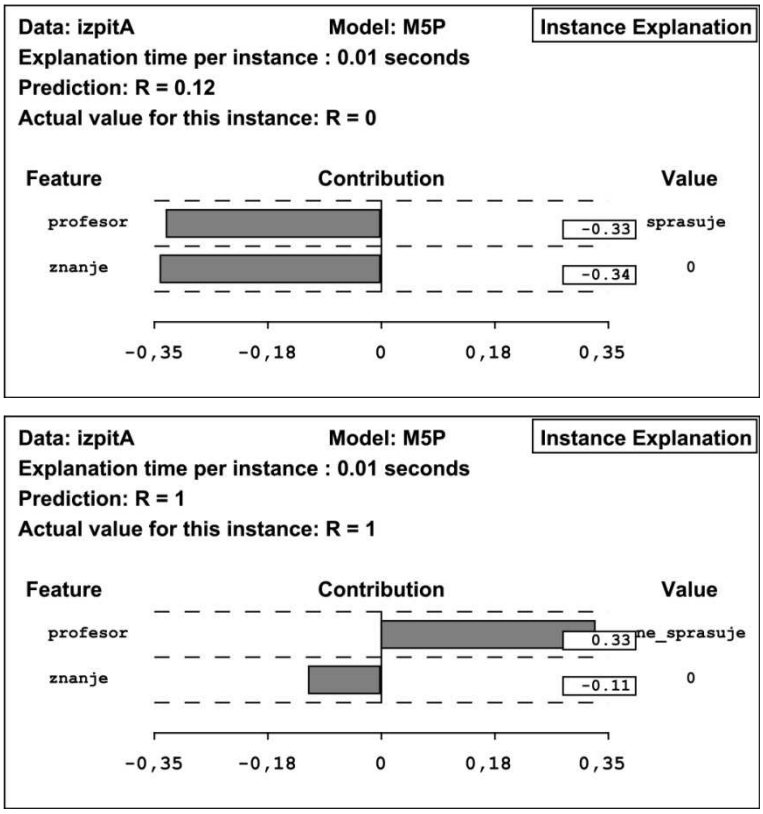
- **Izpit A** - verjetnost da profesor sprašuje je 50%, učenje je enakomerno porazdeljeno.
- **Izpit B** - verjetnost da profesor sprašuje je 90%, učenje je enakomerno porazdeljeno.
- **Izpit C** - verjetnost da profesor sprašuje je 10%, učenje je enakomerno porazdeljeno.
- **Izpit D** - študent se praviloma dobro pripravi na izpit.
- **Izpit E** - študent se praviloma slabo pripravi na izpit.

Porazdelitev vpliva na razlago napovedi, četudi je formula za določitev regresijske spremenljivke enaka za vseh 5 domen. Za vsako izmed domen so napovedovalni algoritmi zgradili podobne modele. Primer za to kaže slika 18.



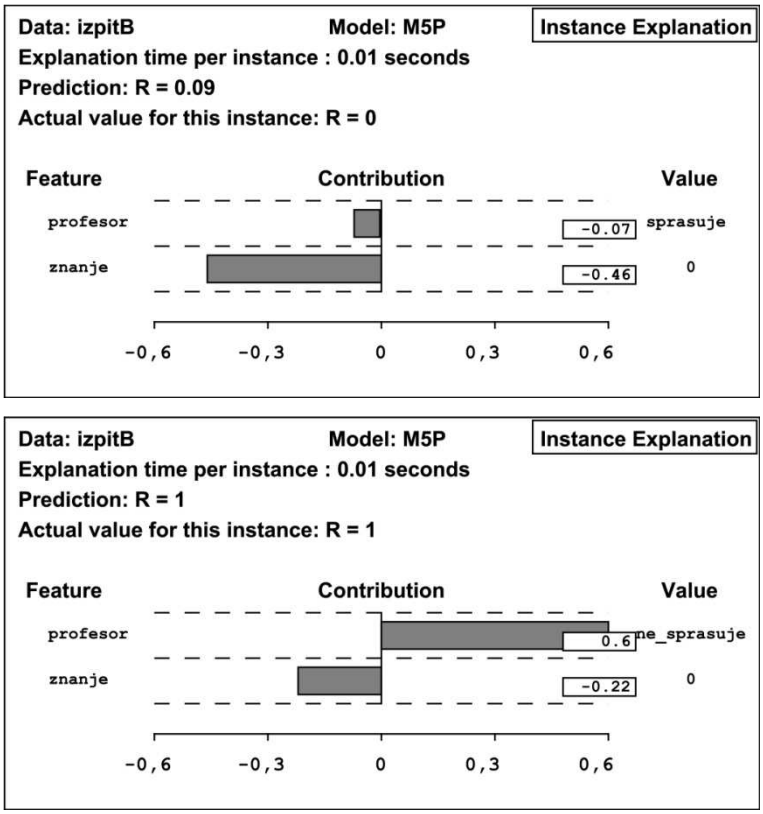
Slika 17: Razlaga napovedi primera nad domeno izpit B za vseh 6 napovedovalnih algoritmov. Vse napovedi so usklajene.

Sedaj si nekoliko podrobneje poglejmo razlage regresijskega drevesa in kako se razlage napovedi spreminjajo glede na distribucijo atributov. Vzeli smo slabo pripravljenega študenta in si ogledali kako se obnašajo prispevki atributov na posameznih izpitih. Primerjali smo dve varianti: v eni na izpitu profesor vpraša študenta, v drugi pa profesor ne sprašuje in študent avtomatično izpit opravi. Kot vidimo na Sliki 18, ko je študent vprašan, zelo negativno vplivata oba atributa. V primeru, kjer ni vprašan, pa atribut znanja, ki sicer še vedno negativno vpliva, zelo malo prispeva, saj močno prevlada dejstvo, da profesor ne sprašuje in znanje na ustnem izpitu takrat izgubi na pomembnosti.



Slika 18: Primer slabo pripravljenega študenta na izpitu A.

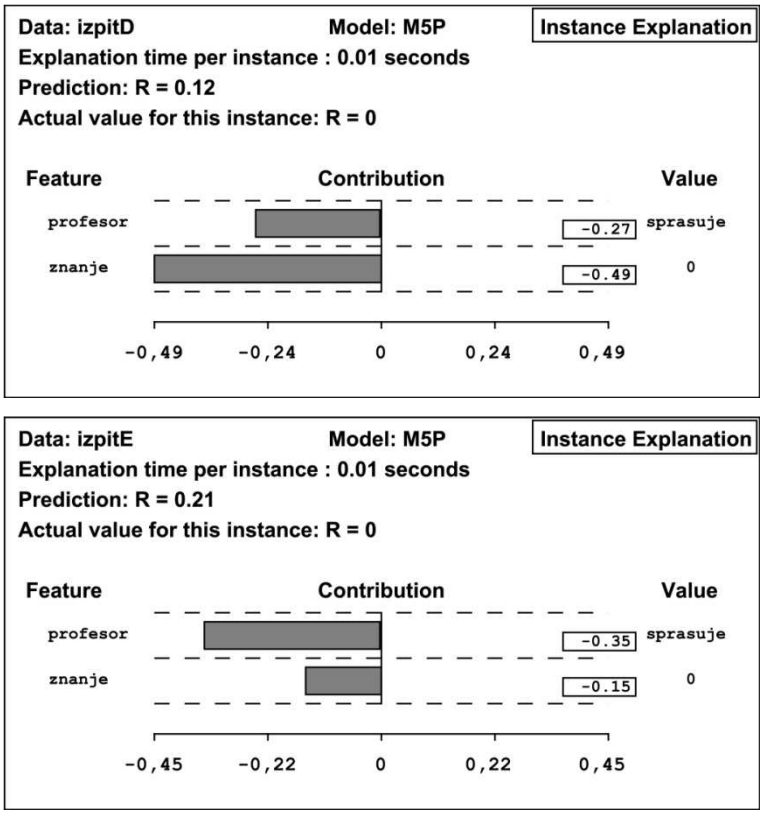
Na izpitu B našemu slabo pripravljenemu študentu slabo kaže, saj profesor večino časa sprašuje. Tako vidimo (glej Slika 19), da v primeru, ko sprašuje, atribut profesor zelo malo vpliva na rezultat, saj večino časa tako ali tako sprašuje. Znanje pa zelo negativno vpliva, ker pri večinskem spraševanju najbolj vpliva znanje. V primeru, ko ne sprašuje, pa atribut profesor pridobi na pomembnosti, saj je glavni dejavnik, da študent izpit opravi - kljub temu da profesor ponavadi sprašuje. Razlaga je torej v skladu z intuitivnim prepričanjem, da je imel študent v tem primeru srečo in izpit opravil predvsem po zaslugi dejstva, da ni bil vprašan.



Slika 19: Primera slabo pripravljenega študenta na izpitu B.

Ker imamo isti primer v dveh različicah, lahko sedaj primerjamo razlagi na izpitu A in izpitu B (sliki 18 in 19). Vidimo, da v primeru, ko profesor na izpitu sprašuje, dobimo povsem enakovredne prispevke obeh atributov na izpitu A. Na izpitu B pa se pomembnost spraševanja prevesi. Tedaj je dejstvo, da profesor sprašuje, precej nepomembno, saj tako ali tako v večini primerov sprašuje. Ko pa profesor ne sprašuje, pa je to izjemen dogodek, ki izstopa od povprečja - in nam s tem pomaga pri določanju regresijske spremenljivke.

Poglejmo si še primer, ki ga kaže Slika 20. Na izpitu E se študent praviloma slabo pripravi, zato dejstvo, da se v tem primeru slabo pripravi, ni tako zelo pomembno kot dejstvo, da pofesor sprašuje. Na izpitu D pa se študent praviloma dobro pripravi. V tem primeru pa neznanje močno odstopa od povprečja, zato tu leži večinski delež prispevka. Vidimo torej, da med dobro pripravljenimi študenti slabo pripravljen študent izstopa (in s tem pridobi neznanje večji vpliv), med slabo pripravljenimi študenti pa ne (neznanje ima manjši vpliv).



Slika 20: Primer slabo pripravljenega študenta na izpitih D in E. V obeh primerih profesor sprašuje in vidna je razlika razlag napovedi zaradi različnih kontekstov.

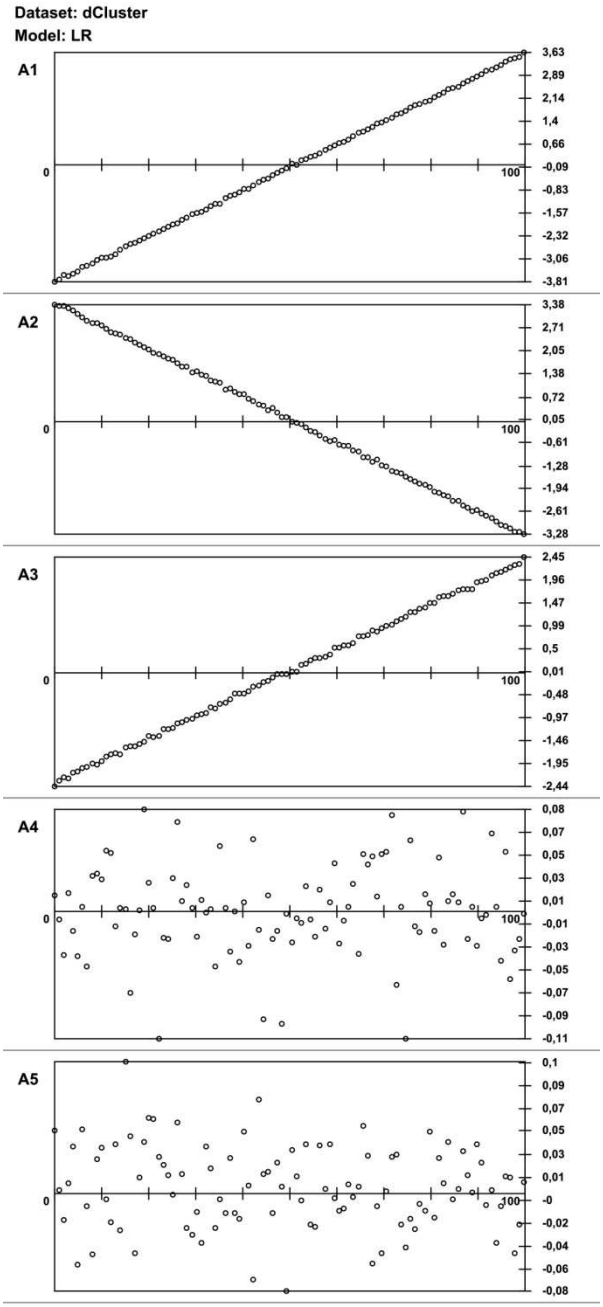
4 ZAKLJUČEK

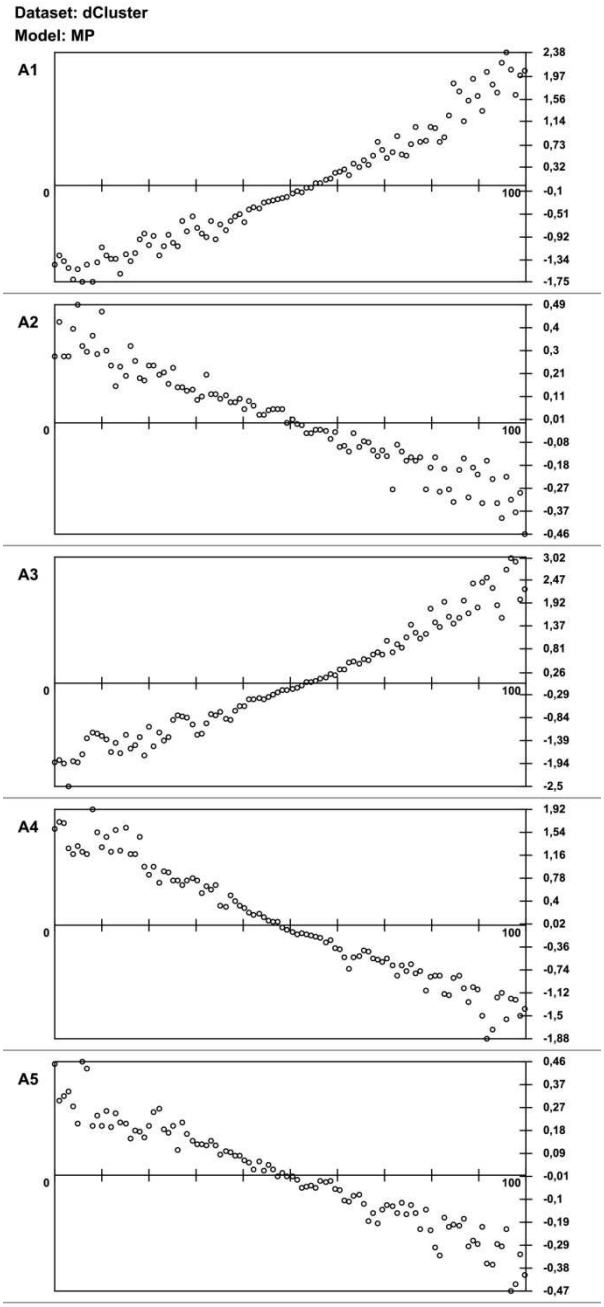
Kot rezultat diplomske naloge lahko zabeležimo uspeh preizkušene metode. Pogledali smo, kako deluje na različnih podatkovnih domenah z različnimi regresijskimi algoritmi in težav nismo srečali. Metoda uspešno razlaga napovedi algoritmov. Pokazali smo, da je kvaliteta napovedi linearno odvisna od kvalitete modela, kar potrjuje uspeh poizkusov. S pomočjo razlage primerov smo uspešno vizualizirali razlage metod, na katerih lahko lepo vidimo način razlage posamezne metode. Poleg tega smo ugotovili, da nam lahko vizualizacije razlag metod pomagajo z različnimi napakami razlag. Primer tega je bil problem z redundantnimi atributi. Težave, ki smo jih srečali, niso bile vezane na metodo razlage, temveč na časovno zahtevnost napovedi določenih algoritmov (v glavnem KNN in SVM), ter na poprej omenjeni problem s redundantnimi atributi, kjer definicija razlage napovedi pravilnega modela v tem primeru dopušča dvoumnost oz. več pravilnih razlag. Z razlago v kontekstu smo pokazali, kako naša razlaga upošteva kontekst in kako pomembno vlogo igra porazdelitev atributa pri razlagi napovedi in temu, kako si ljudje razlagamo primere.

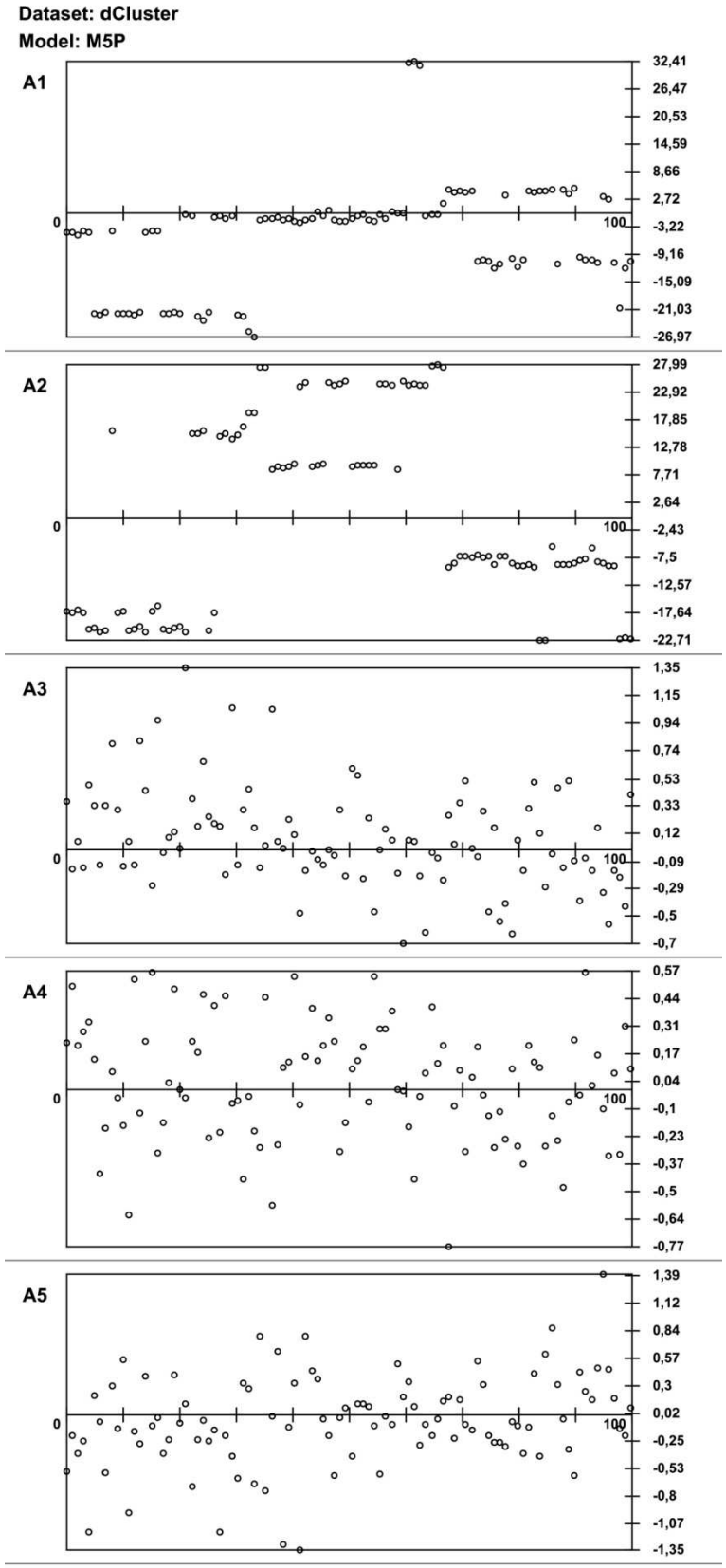
Kot nadaljnje delo se nam ponuja časovna zahtevnost trenutne metode. Število vzorcev bi lahko določili dinamično in ne tako, kot v tej diplomski nalogi, vnaprej. Število vzorcev bi lahko bilo tudi odvisno od primera, ki ga rešujemo ali pa napovedovalnega algoritma (podobno smo morali v nalogi ukrepati z obema metodama KNN). Glavni cilj je minimizacija števila vzorcev za doseg določene maksimalne velikosti napake. Poleg časovne zahtevnosti je problem pri redundantnih atributih in definiciji razlage napovedi pravilnega modela. Poraja se vprašanje: Ali kot razlago napovedi pravilnega modela dopuščamo samo eno razlago ali jih je lahko več? V primeru, da je pravilna le ena razlaga, katera razlaga je potem pravilna? Tista, ki upošteva enako pomembnost atributov ali tista, ki zanemari enega od dveh atributov? Če zanemarimo en atribut - katerega? Problem se pojavi tudi, če imamo več pravilnih razlag, kako sedaj izračunamo razdaljo do pravilne rešitve. Tu se ponuja več opcij, kot so povprečje vseh dopustnih rešitev, najmanjša od vseh dopustnih povprečij, ipd. Poleg omenjenih možnosti za nadaljnje delo lahko omenimo še splošno možnost drugotnih pohitritev (dostop dopušča veliko možnost paralelnega izvajanja) ter preizkus metode na večjem spektru regresijskih algoritmov.

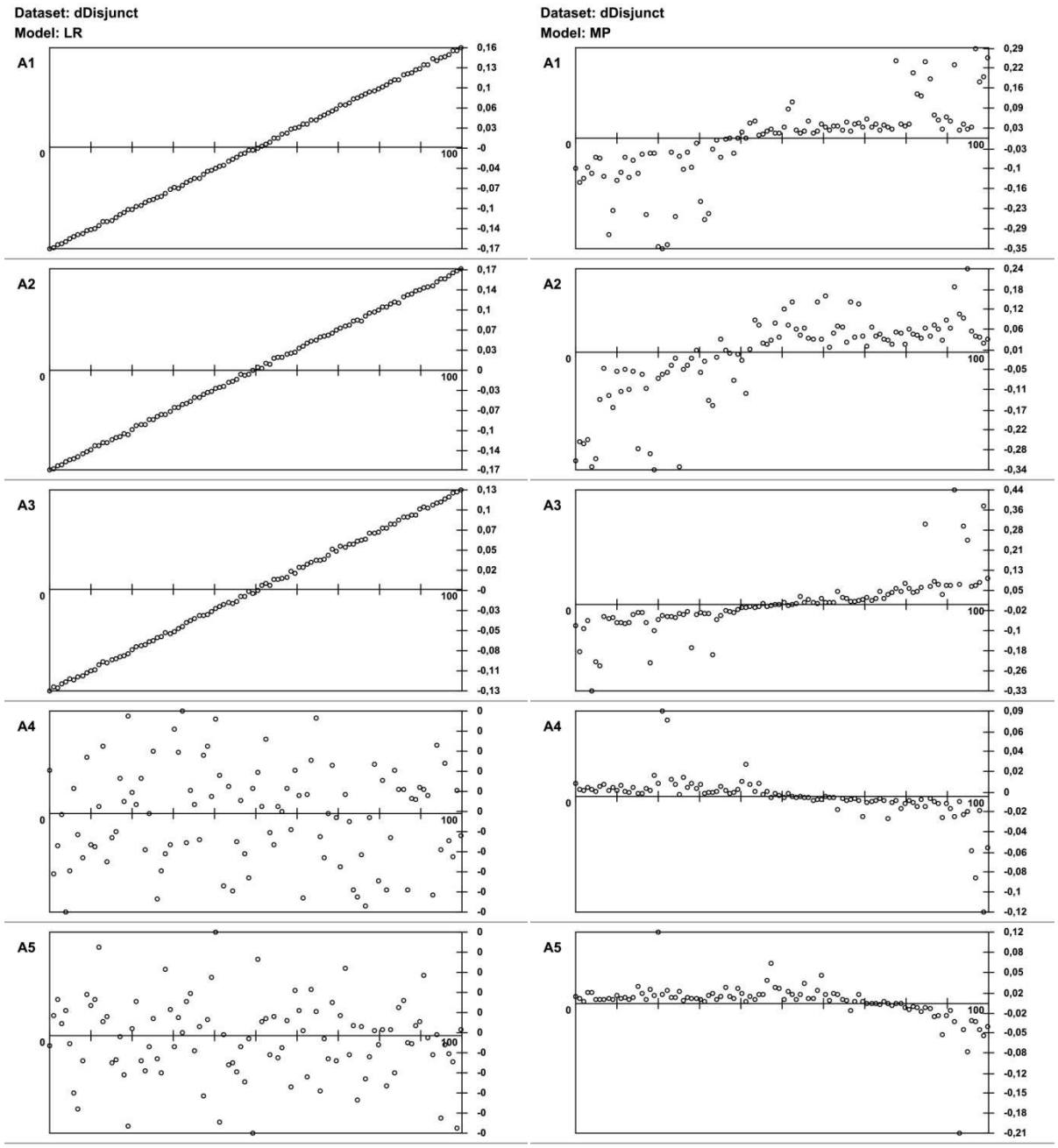
Dodatek A

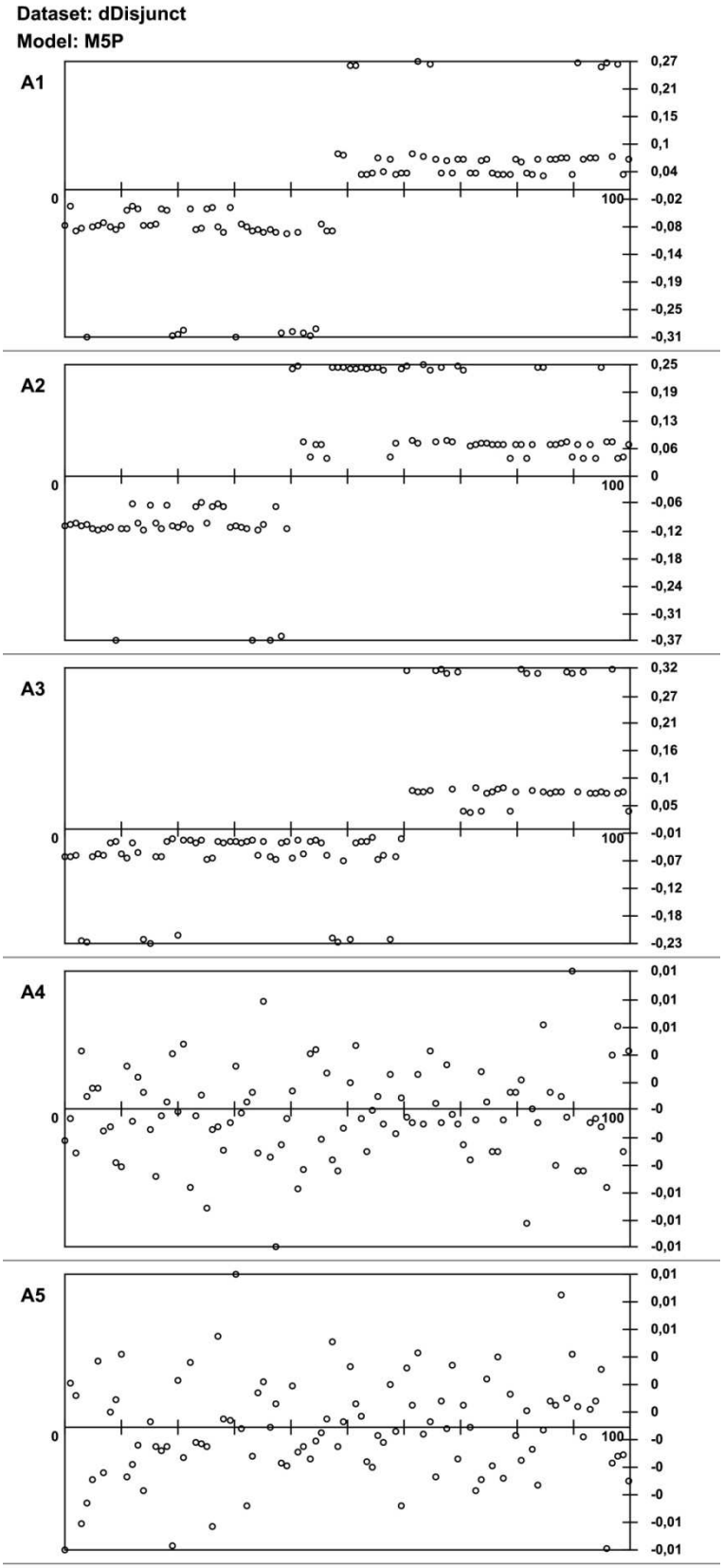
RAZLAGE MODELOV LR, M5P IN MP

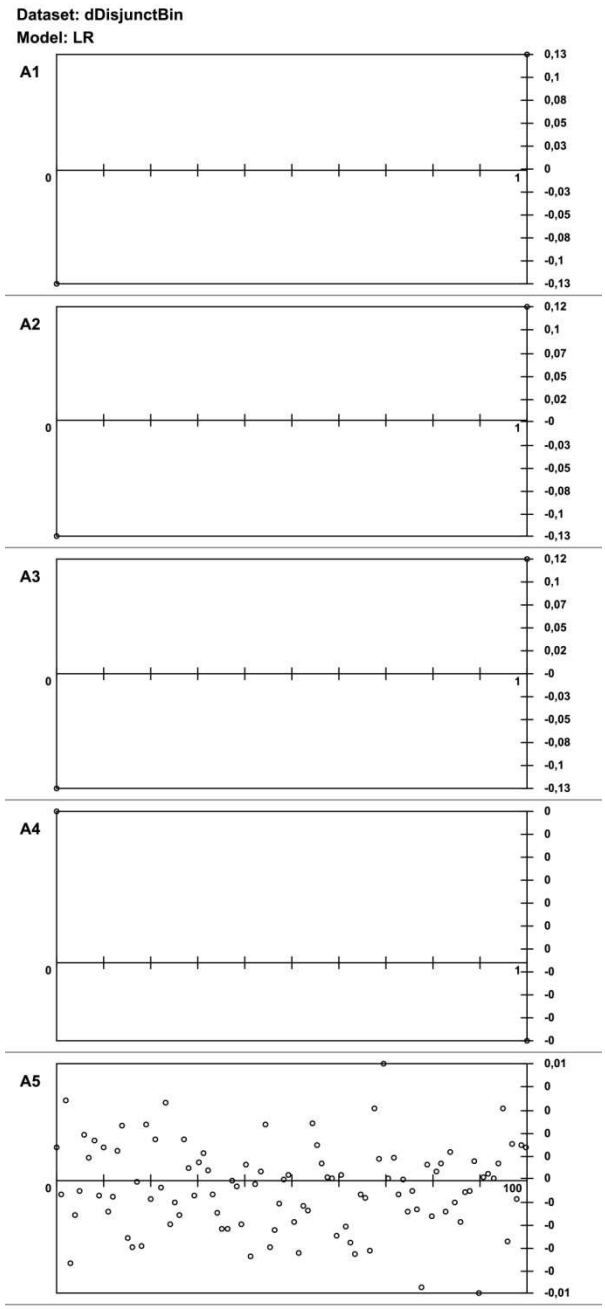


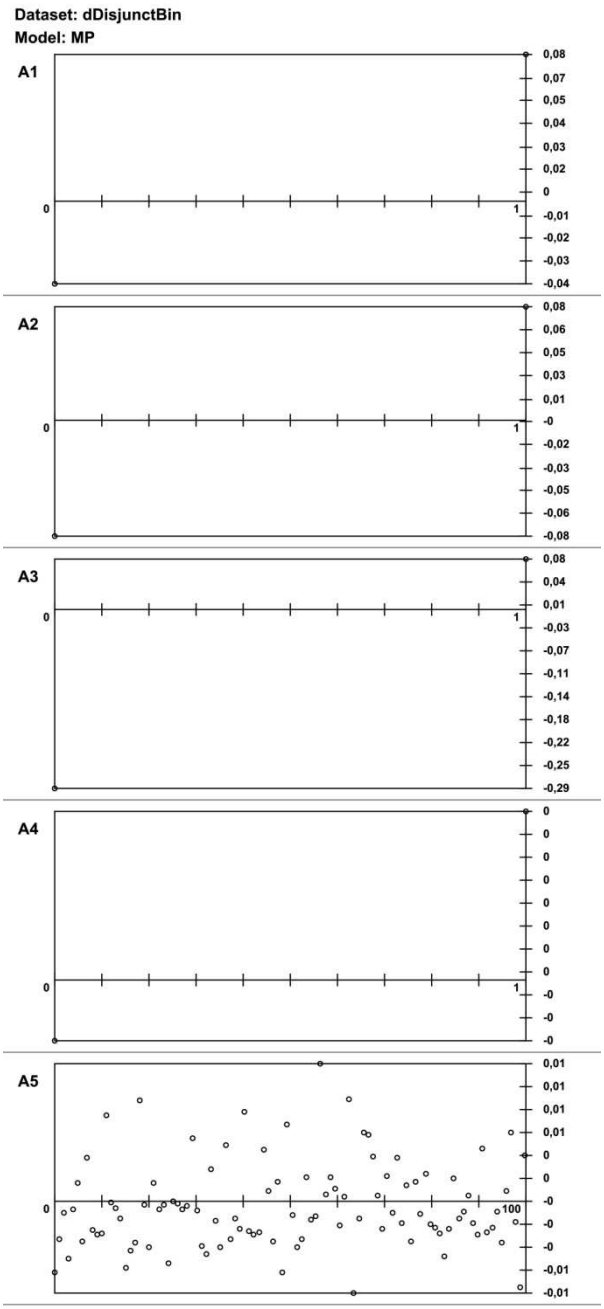


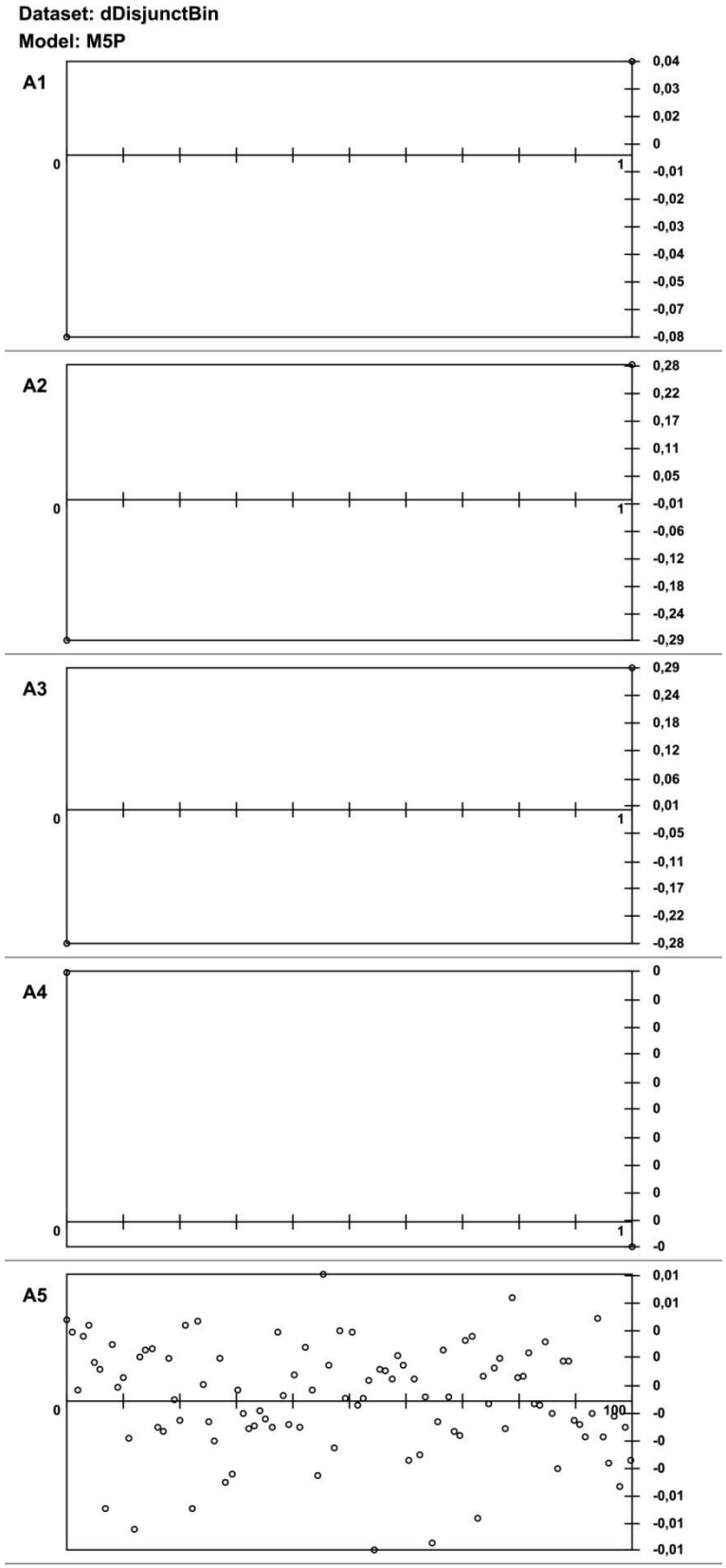

Dataset: dCluster
Model: M5P
A1
32,41
26,47
20,53
14,59
8,66
2,72
-3,22
-9,16
-15,09
-21,03
-26,97
0
100
A2
27,99
22,92
17,85
12,78
7,71
2,64
-2,43
-7,5
-12,57
-17,64
-22,71
0
100
A3
1,35
1,15
0,94
0,74
0,53
0,33
0,12
-0,09
-0,29
-0,5
-0,7
0
100
A4
0,57
0,44
0,31
0,17
0,04
-0,1
-0,23
-0,37
-0,5
-0,64
-0,77
0
100
A5
1,39
1,12
0,84
0,57
0,3
0,02
-0,25
-0,53
-0,8
-1,07
-1,35
0
100

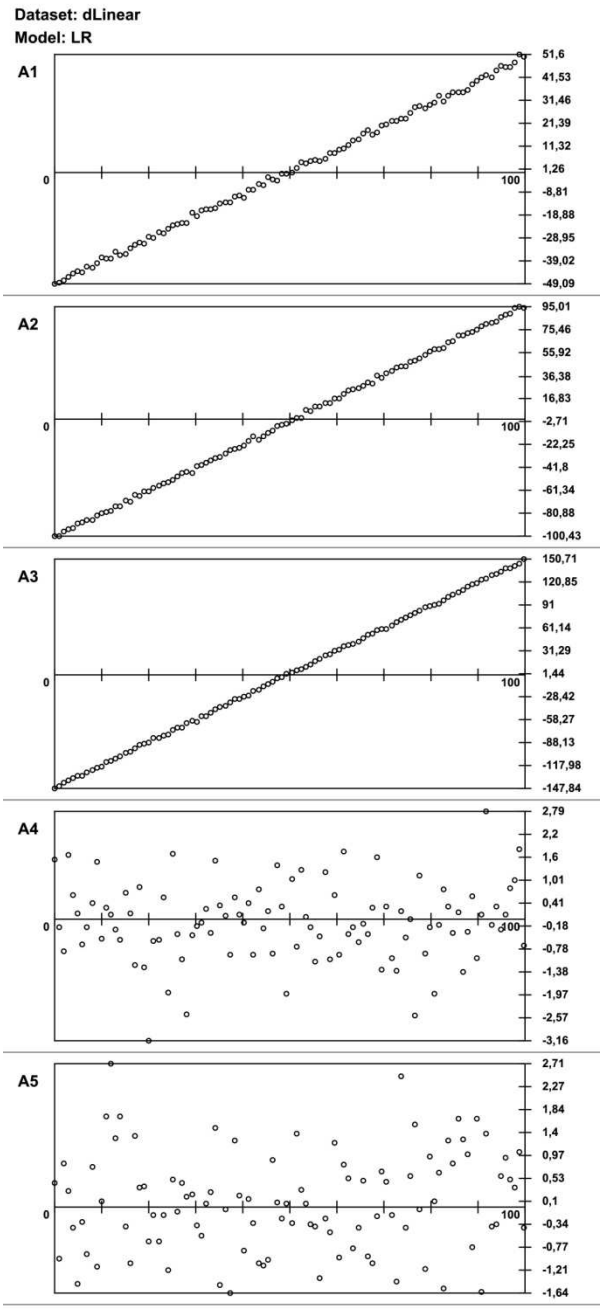

Dataset: dDisjunct
Model: LR
A1
A2
A3
A4
A5
Dataset: dDisjunct
Model: MP
A1
A2
A3
A4
A5

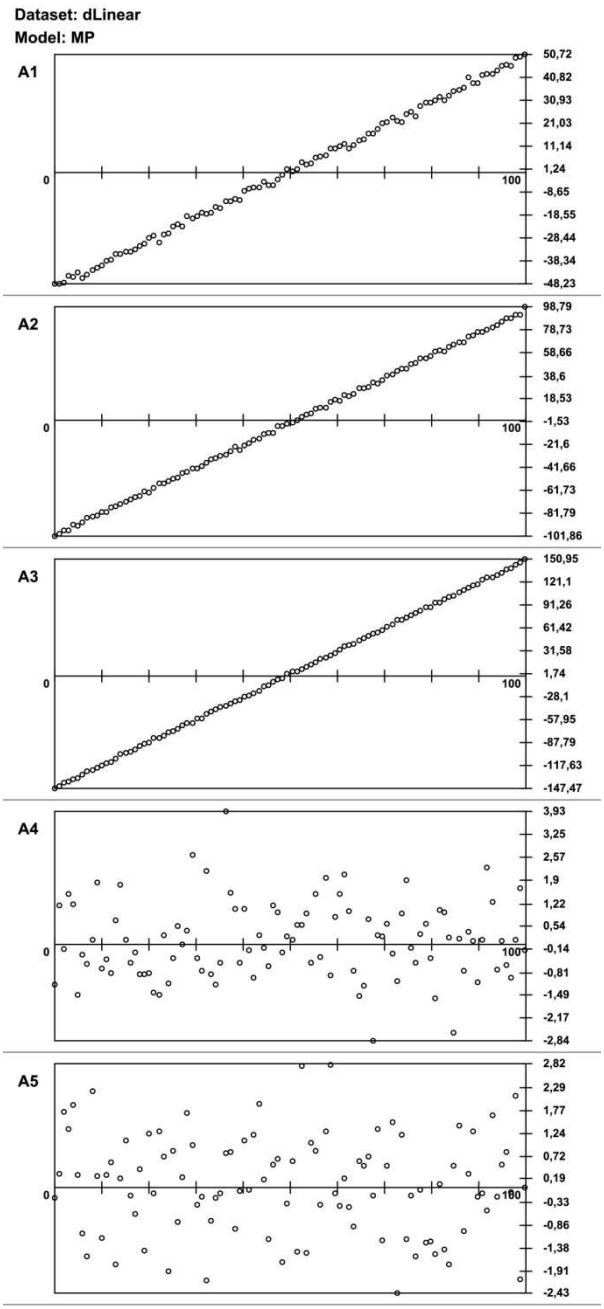

Dataset: dDisjunct
Model: M5P
A1
0,27
0,21
0,15
0,1
0,04
-0,02
-0,08
-0,14
-0,19
-0,25
-0,31
0
100
A2
0,25
0,19
0,13
0,06
0
-0,06
-0,12
-0,18
-0,24
-0,31
-0,37
0
100
A3
0,32
0,27
0,21
0,16
0,1
0,05
-0,01
-0,07
-0,12
-0,18
-0,23
0
100
A4
0,01
0,01
0,01
0
0
-0
-0
-0
-0,01
-0,01
-0,01
0
100
A5
0,01
0,01
0,01
0
0
0
-0
-0
-0
-0,01
-0,01
0
100

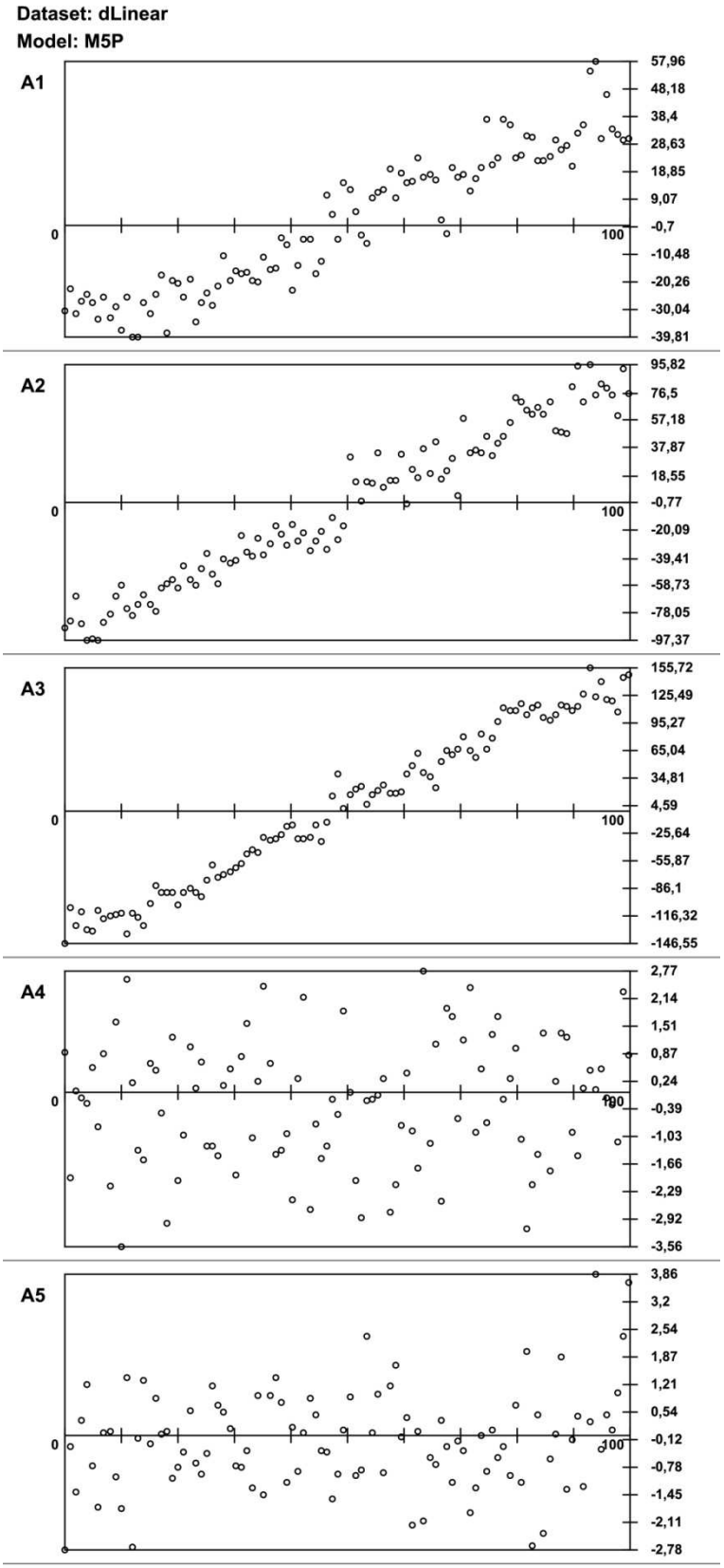

Dataset: dDisjunctBin
Model: LR
A1
A2
A3
A4
A5

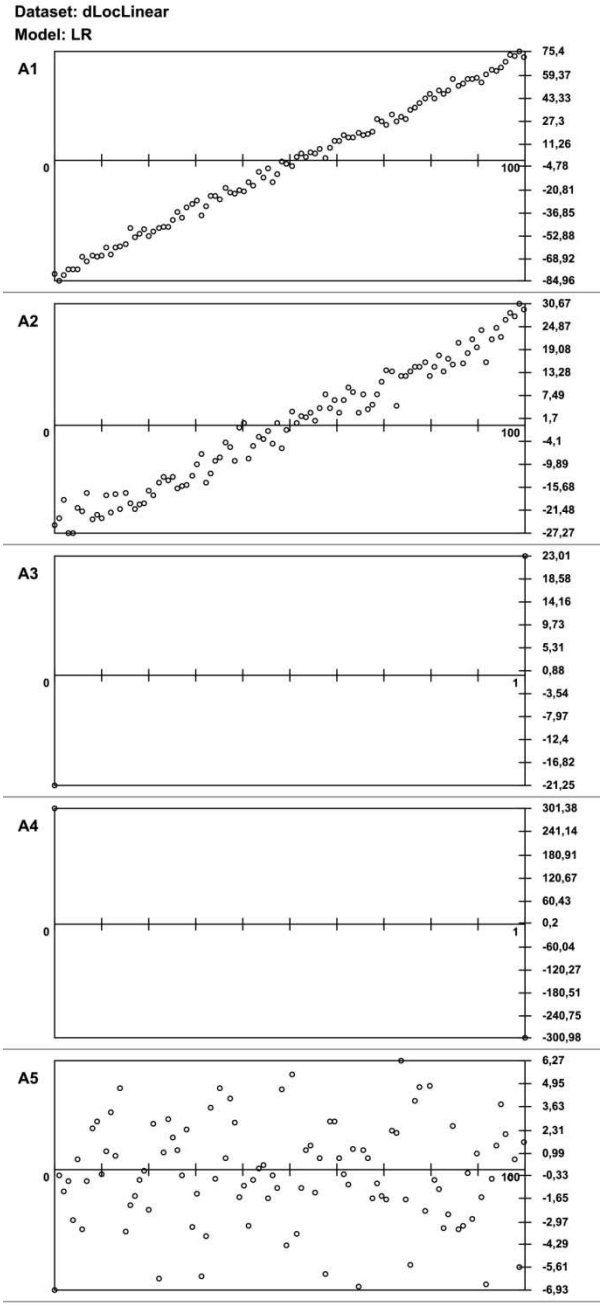

Dataset: dDisjunctBin
Model: MP
A1
A2
A3
A4
A5

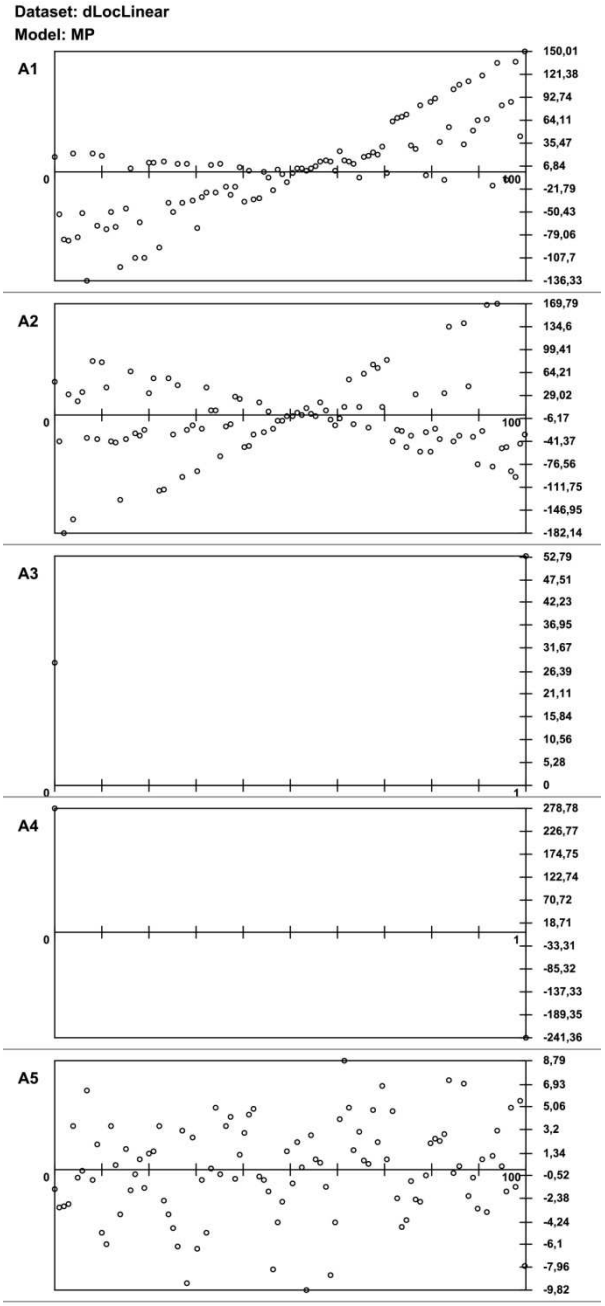

Dataset: dDisjunctBin
Model: M5P
A1
0
1
0,04
0,03
0,02
0
-0,01
-0,02
-0,03
-0,04
-0,05
-0,07
-0,08
A2
0
1
0,28
0,22
0,17
0,11
0,05
-0,01
-0,06
-0,12
-0,18
-0,24
-0,29
A3
0
1
0,29
0,24
0,18
0,12
0,06
0,01
-0,05
-0,11
-0,17
-0,22
-0,28
A4
0
1
0
0
0
0
0
0
0
0
0
0
-0
A5
0
100
0,01
0,01
0
0
0
-0
-0
-0
-0,01
-0,01
-0,01

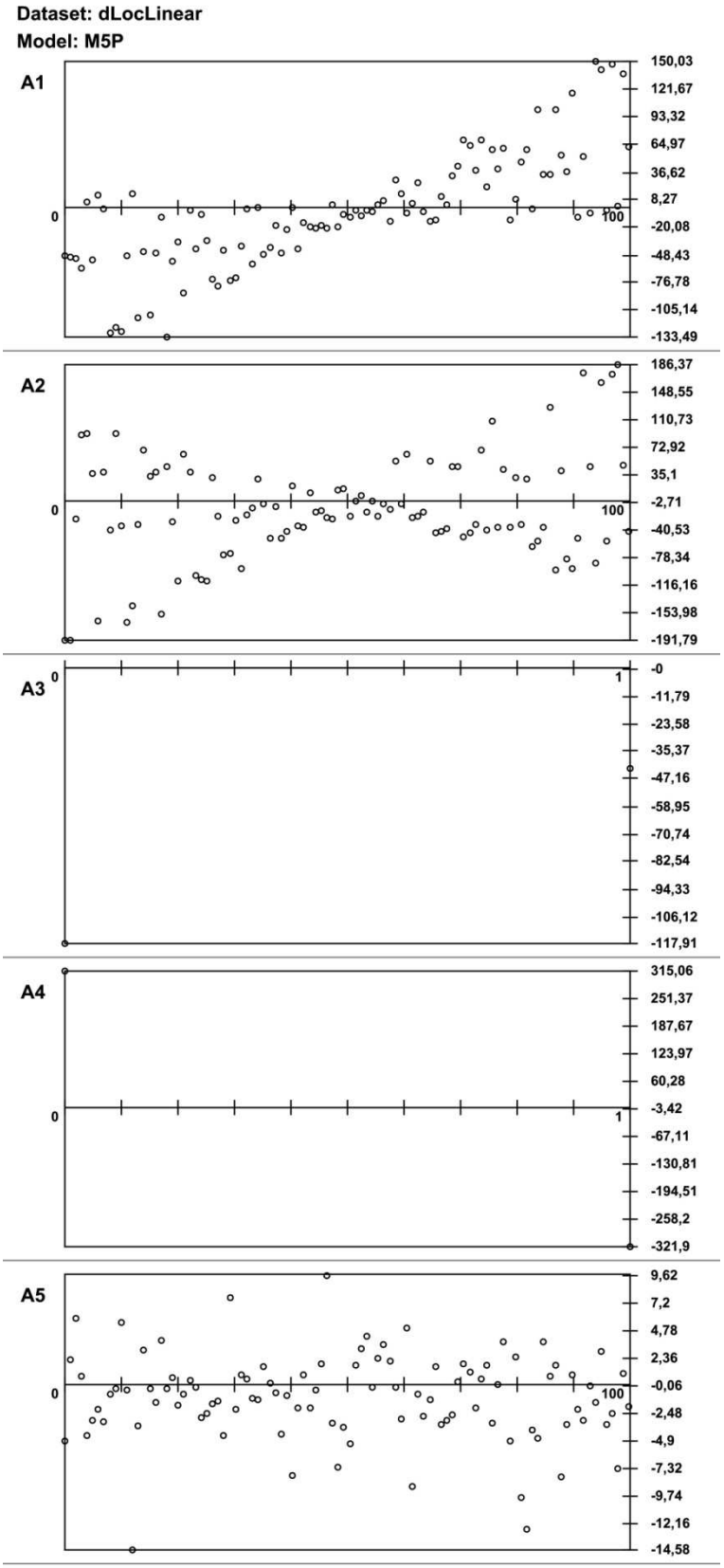

Dataset: dLinear
Model: LR
A1
51,6
41,53
31,46
21,39
11,32
1,26
-8,81
-18,88
-28,95
-39,02
-49,09
0
100
A2
95,01
75,46
55,92
36,38
16,83
-2,71
-22,25
-41,8
-61,34
-80,88
-100,43
A3
150,71
120,85
91
61,14
31,29
1,44
-28,42
-58,27
-88,13
-117,98
-147,84
A4
2,79
2,2
1,6
1,01
0,41
-0,18
-0,78
-1,38
-1,97
-2,57
-3,16
A5
2,71
2,27
1,84
1,4
0,97
0,53
0,1
-0,34
-0,77
-1,21
-1,64

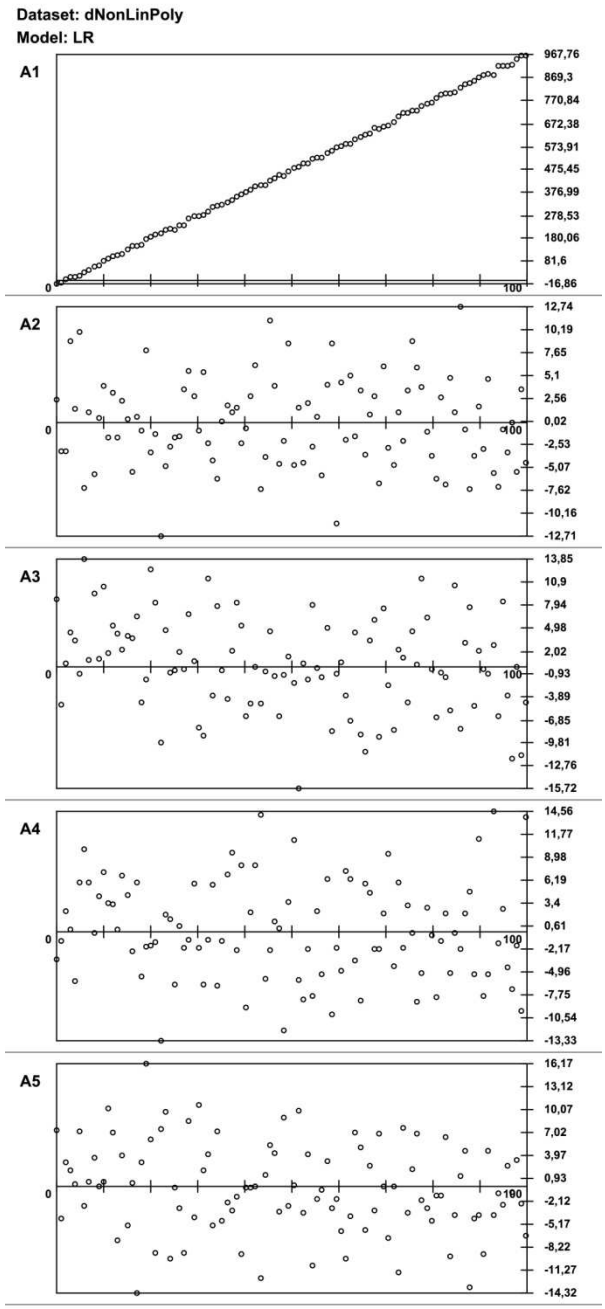

Dataset: dLinear
Model: MP
A1
50,72
40,82
30,93
21,03
11,14
1,24
-8,65
-18,55
-28,44
-38,34
-48,23
0
100
A2
98,79
78,73
58,66
38,6
18,53
-1,53
-21,6
-41,66
-61,73
-81,79
-101,86
A3
150,95
121,1
91,26
61,42
31,58
1,74
-28,1
-57,95
-87,79
-117,63
-147,47
A4
3,93
3,25
2,57
1,9
1,22
0,54
-0,14
-0,81
-1,49
-2,17
-2,84
A5
2,82
2,29
1,77
1,24
0,72
0,19
-0,33
-0,86
-1,38
-1,91
-2,43

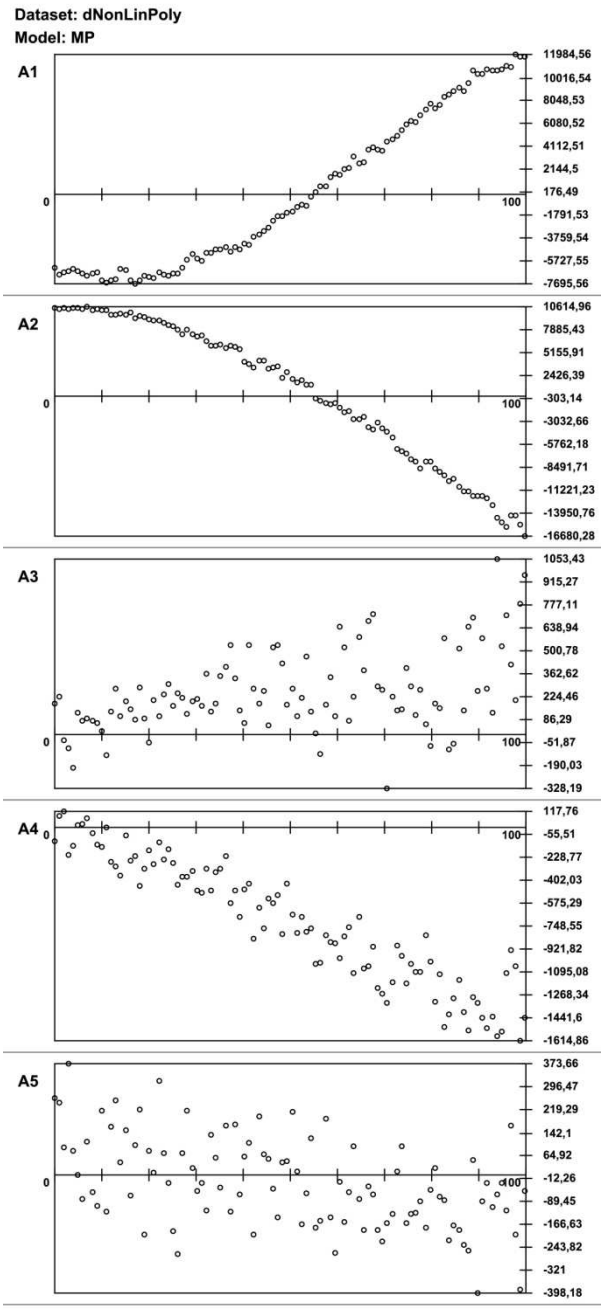

Dataset: dLinear
Model: M5P
A1
57,96
48,18
38,4
28,63
18,85
9,07
-0,7
-10,48
-20,26
-30,04
-39,81
0
100
A2
95,82
76,5
57,18
37,87
18,55
-0,77
-20,09
-39,41
-58,73
-78,05
-97,37
0
100
A3
155,72
125,49
95,27
65,04
34,81
4,59
-25,64
-55,87
-86,1
-116,32
-146,55
0
100
A4
2,77
2,14
1,51
0,87
0,24
-0,39
-1,03
-1,66
-2,29
-2,92
-3,56
0
100
A5
3,86
3,2
2,54
1,87
1,21
0,54
-0,12
-0,78
-1,45
-2,11
-2,78
0
100

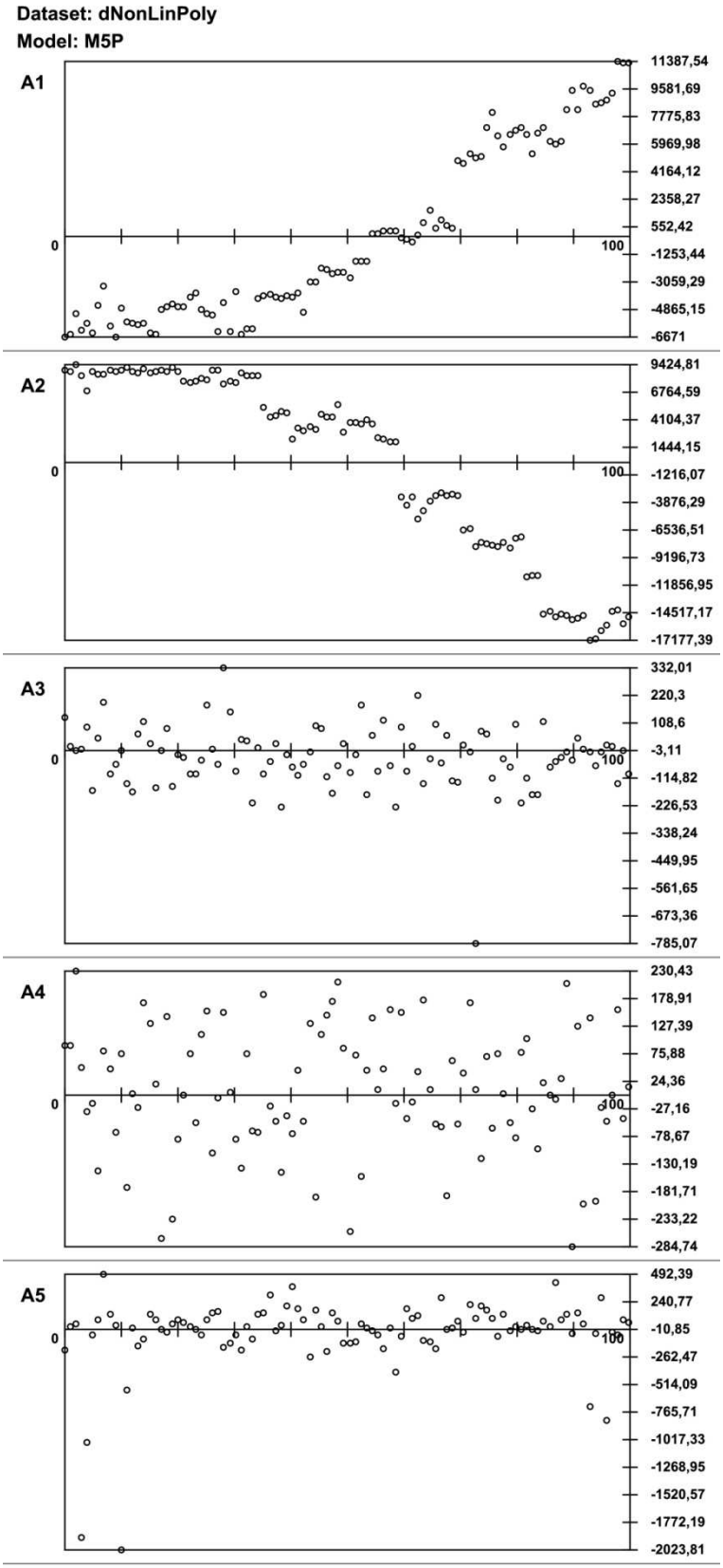

Dataset: dLocLinear
Model: LR
A1
75,4
59,37
43,33
27,3
11,26
-4,78
-20,81
-36,85
-52,88
-68,92
-84,96
0
100
A2
30,67
24,87
19,08
13,28
7,49
1,7
-4,1
-9,89
-15,68
-21,48
-27,27
A3
23,01
18,58
14,16
9,73
5,31
0,88
-3,54
-7,97
-12,4
-16,82
-21,25
1
A4
301,38
241,14
180,91
120,67
60,43
0,2
-60,04
-120,27
-180,51
-240,75
-300,98
A5
6,27
4,95
3,63
2,31
0,99
-0,33
-1,65
-2,97
-4,29
-5,61
-6,93

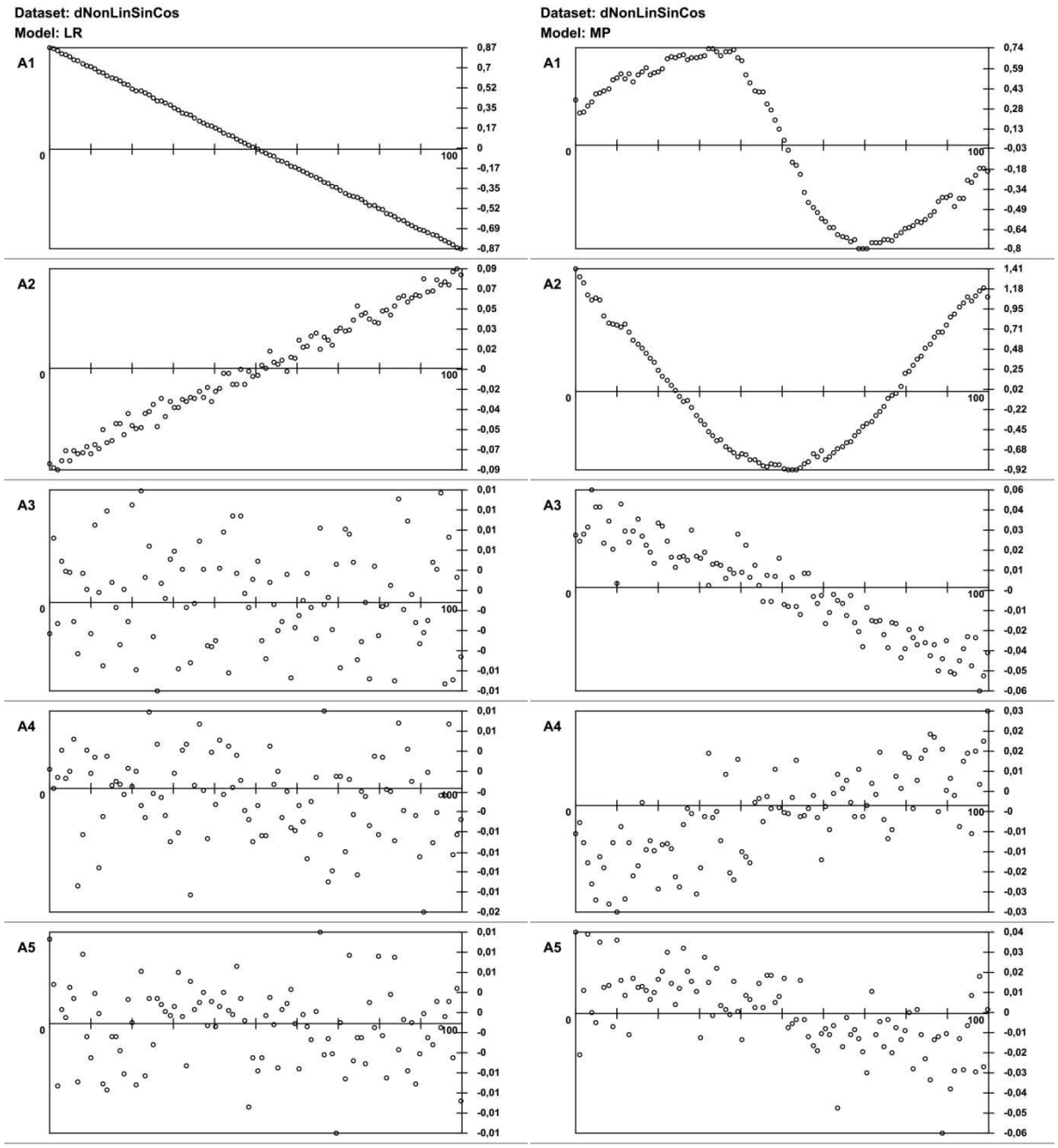

Dataset: dLocLinear
Model: MP
A1
150,01
121,38
92,74
64,11
35,47
6,84
-21,79
-50,43
-79,06
-107,7
-136,33
0
100
A2
169,79
134,6
99,41
64,21
29,02
-6,17
-41,37
-76,56
-111,75
-146,95
-182,14
A3
52,79
47,51
42,23
36,95
31,67
26,39
21,11
15,84
10,56
5,28
0
1
A4
278,78
226,77
174,75
122,74
70,72
18,71
-33,31
-85,32
-137,33
-189,35
-241,36
A5
8,79
6,93
5,06
3,2
1,34
-0,52
-2,38
-4,24
-6,1
-7,96
-9,82

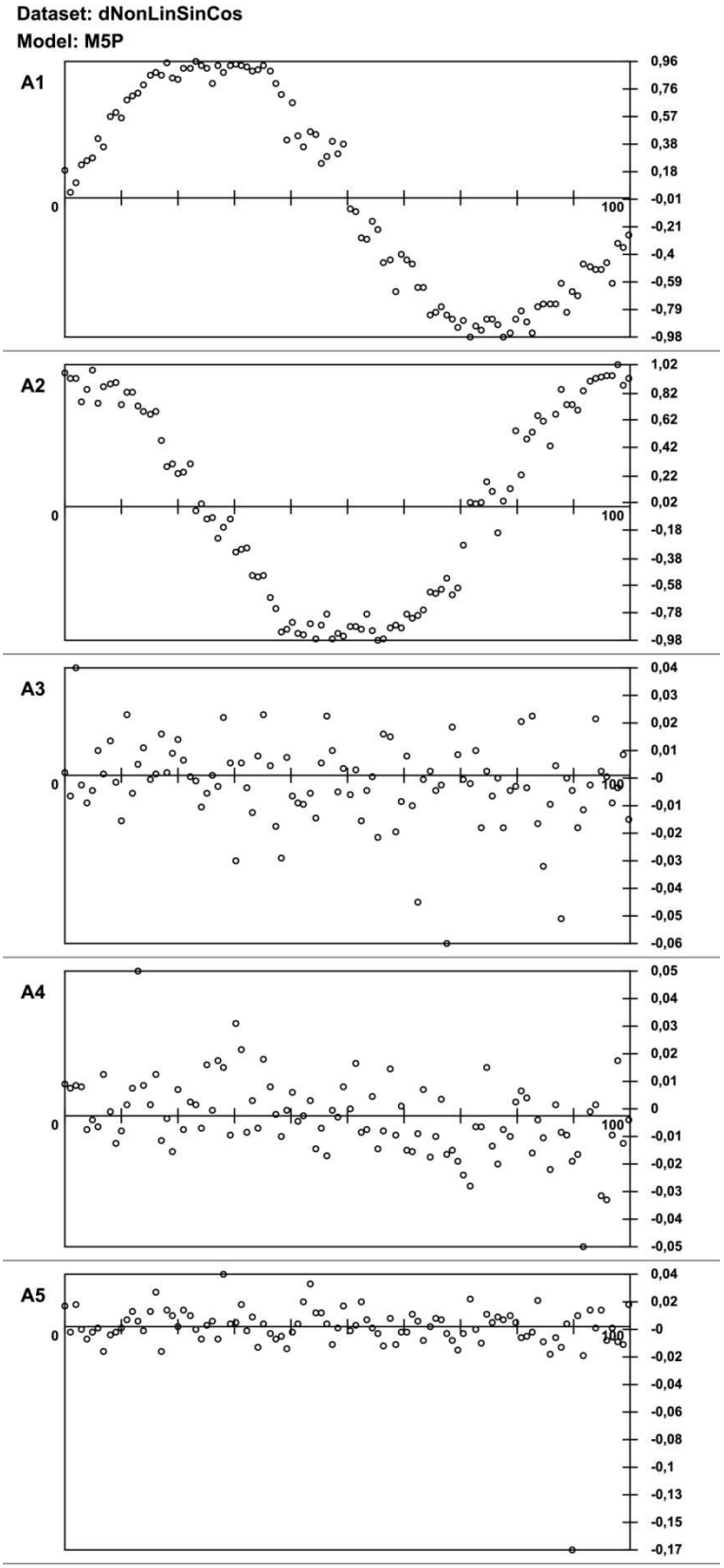

Dataset: dLocLinear
Model: M5P
A1
150,03
121,67
93,32
64,97
36,62
8,27
-20,08
-48,43
-76,78
-105,14
-133,49
0
100
A2
186,37
148,55
110,73
72,92
35,1
-2,71
-40,53
-78,34
-116,16
-153,98
-191,79
0
100
A3
-0
-11,79
-23,58
-35,37
-47,16
-58,95
-70,74
-82,54
-94,33
-106,12
-117,91
0
1
A4
315,06
251,37
187,67
123,97
60,28
-3,42
-67,11
-130,81
-194,51
-258,2
-321,9
0
1
A5
9,62
7,2
4,78
2,36
-0,06
-2,48
-4,9
-7,32
-9,74
-12,16
-14,58
0
100

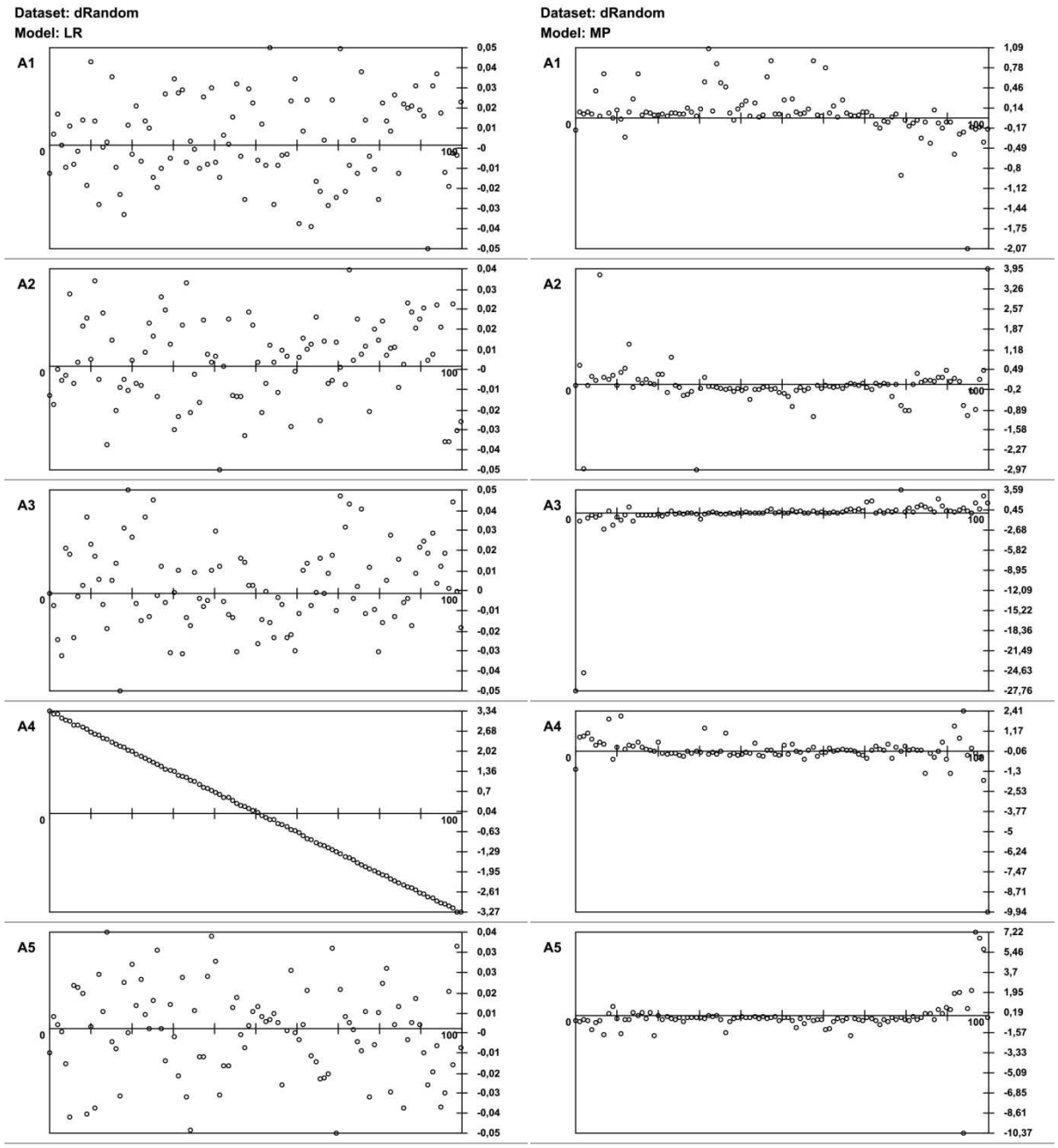

Dataset: dNonLinPoly
Model: LR
A1
967,76
869,3
770,84
672,38
573,91
475,45
376,99
278,53
180,06
81,6
-16,86
0
100
A2
12,74
10,19
7,65
5,1
2,56
0,02
-2,53
-5,07
-7,62
-10,16
-12,71
A3
13,85
10,9
7,94
4,98
2,02
-0,93
-3,89
-6,85
-9,81
-12,76
-15,72
A4
14,56
11,77
8,98
6,19
3,4
0,61
-2,17
-4,96
-7,75
-10,54
-13,33
A5
16,17
13,12
10,07
7,02
3,97
0,93
-2,12
-5,17
-8,22
-11,27
-14,32

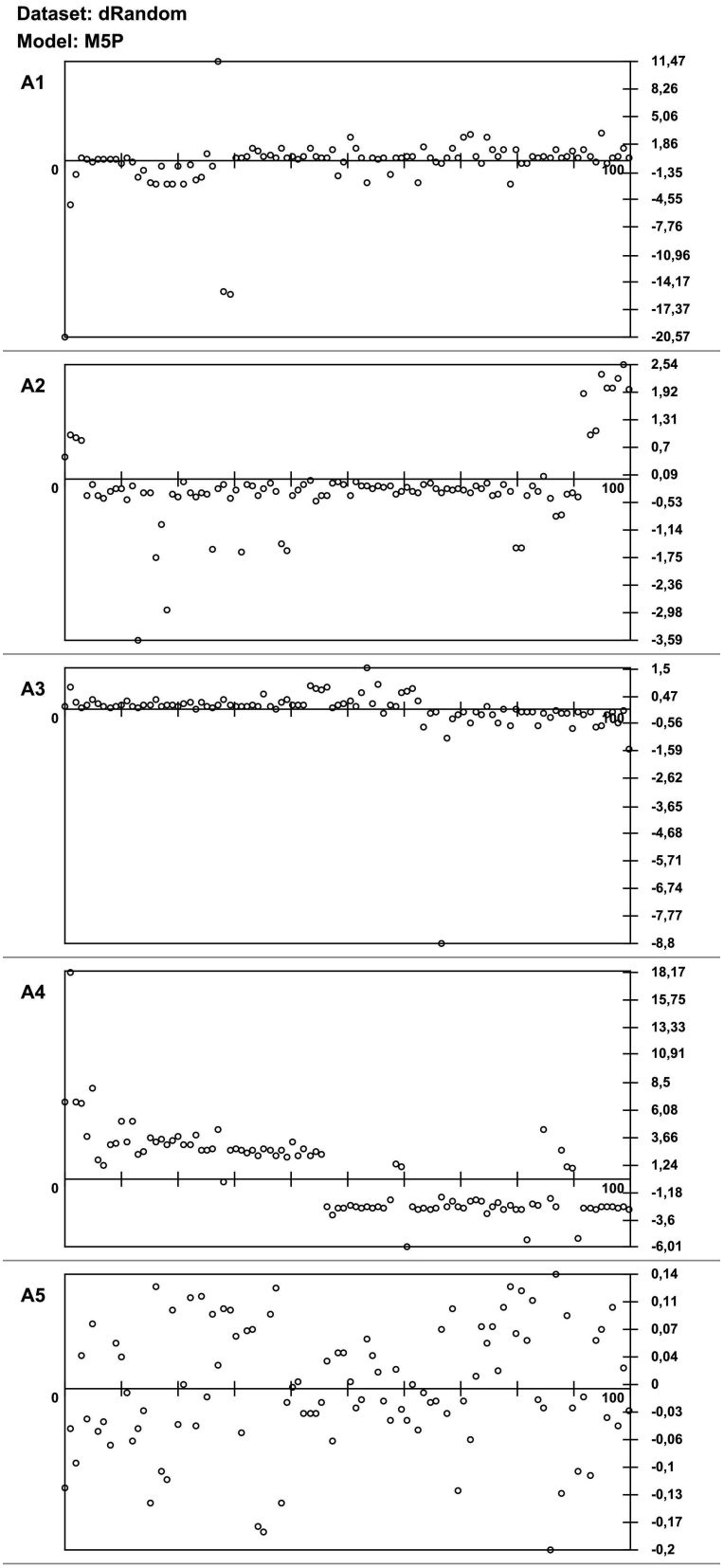

Dataset: dNonLinPoly
Model: MP
A1
11984,56
10016,54
8048,53
6080,52
4112,51
2144,5
176,49
-1791,53
-3759,54
-5727,55
-7695,56
0
100
A2
10614,96
7885,43
5155,91
2426,39
-303,14
-3032,66
-5762,18
-8491,71
-11221,23
-13950,76
-16680,28
A3
1053,43
915,27
777,11
638,94
500,78
362,62
224,46
86,29
-51,87
-190,03
-328,19
A4
117,76
-55,51
-228,77
-402,03
-575,29
-748,55
-921,82
-1095,08
-1268,34
-1441,6
-1614,86
A5
373,66
296,47
219,29
142,1
64,92
-12,26
-89,45
-166,63
-243,82
-321
-398,18

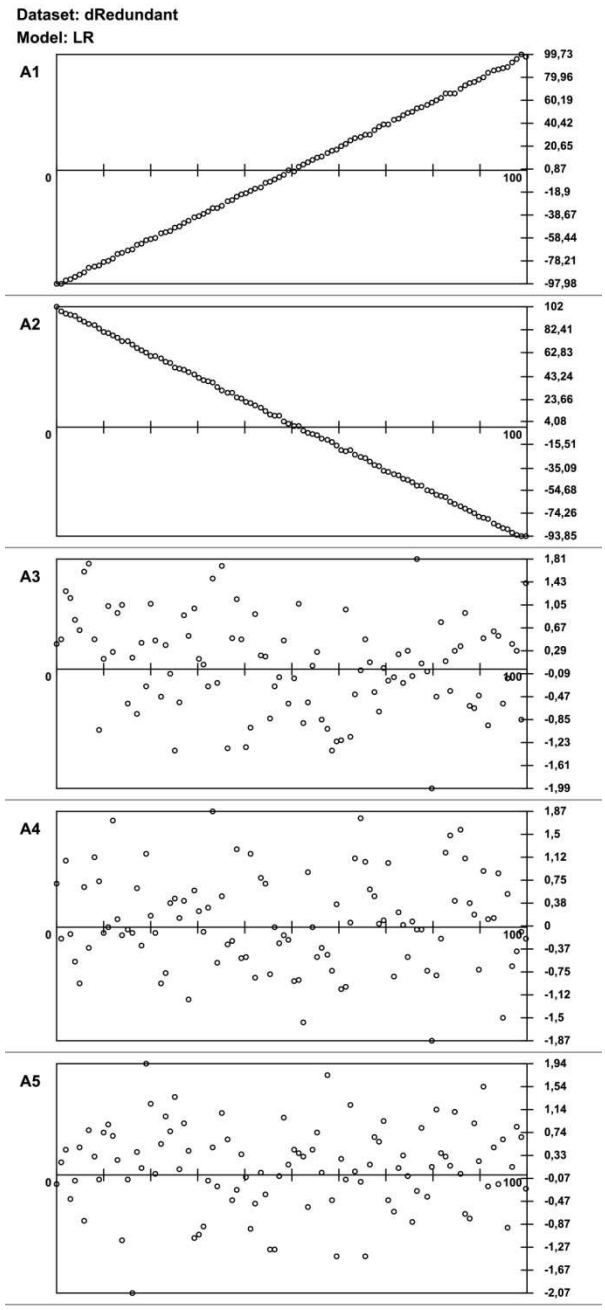

Dataset: dNonLinPoly
Model: M5P
A1
11387,54
9581,69
7775,83
5969,98
4164,12
2358,27
552,42
-1253,44
-3059,29
-4865,15
-6671
0
100
A2
9424,81
6764,59
4104,37
1444,15
-1216,07
-3876,29
-6536,51
-9196,73
-11856,95
-14517,17
-17177,39
A3
332,01
220,3
108,6
-3,11
-114,82
-226,53
-338,24
-449,95
-561,65
-673,36
-785,07
A4
230,43
178,91
127,39
75,88
24,36
-27,16
-78,67
-130,19
-181,71
-233,22
-284,74
A5
492,39
240,77
-10,85
-262,47
-514,09
-765,71
-1017,33
-1268,95
-1520,57
-1772,19
-2023,81

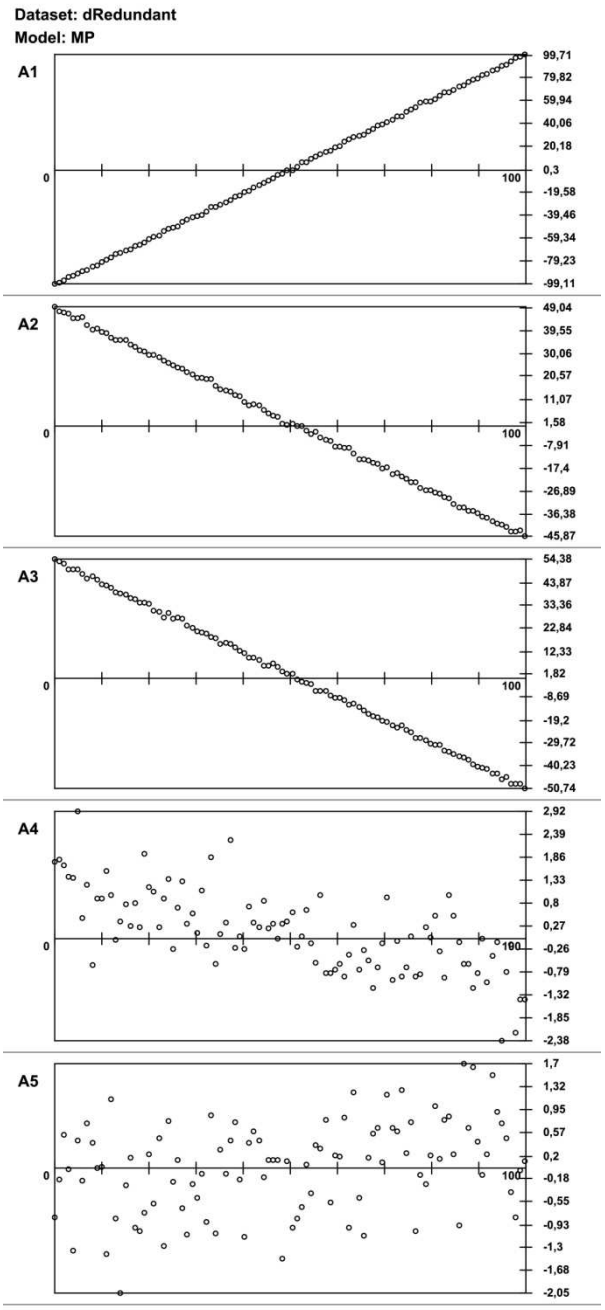

Dataset: dNonLinSinCos
Model: LR
A1
0,87 0,7 0,52 0,35 0,17 0 -0,17 -0,35 -0,52 -0,69 -0,87
0
100
A2
0,09 0,07 0,05 0,03 0,02 -0 -0,02 -0,04 -0,05 -0,07 -0,09
0
100
A3
0,01 0,01 0,01 0,01 0 0 -0 -0 -0 -0,01 -0,01
0
100
A4
0,01 0,01 0 0 -0 -0 -0,01 -0,01 -0,01 -0,01 -0,02
0
100
A5
0,01 0,01 0,01 0 0 -0 -0 -0,01 -0,01 -0,01 -0,01
0
100
Dataset: dNonLinSinCos
Model: MP
A1
0,74 0,59 0,43 0,28 0,13 -0,03 -0,18 -0,34 -0,49 -0,64 -0,8
0
100
A2
1,41 1,18 0,95 0,71 0,48 0,25 0,02 -0,22 -0,45 -0,68 -0,92
0
100
A3
0,06 0,04 0,03 0,02 0,01 -0 -0,01 -0,02 -0,04 -0,05 -0,06
0
100
A4
0,03 0,02 0,02 0,01 0 -0 -0,01 -0,01 -0,02 -0,03 -0,03
0
100
A5
0,04 0,03 0,02 0,01 0 -0,01 -0,02 -0,03 -0,04 -0,05 -0,06
0
100


Dataset: dNonLinSinCos
Model: M5P
A1
0,96
0,76
0,57
0,38
0,18
-0,01
-0,21
-0,4
-0,59
-0,79
-0,98
0
100
A2
1,02
0,82
0,62
0,42
0,22
0,02
-0,18
-0,38
-0,58
-0,78
-0,98
0
100
A3
0,04
0,03
0,02
0,01
-0
-0,01
-0,02
-0,03
-0,04
-0,05
-0,06
0
100
A4
0,05
0,04
0,03
0,02
0,01
0
-0,01
-0,02
-0,03
-0,04
-0,05
0
100
A5
0,04
0,02
-0
-0,02
-0,04
-0,06
-0,08
-0,1
-0,13
-0,15
-0,17
0
100

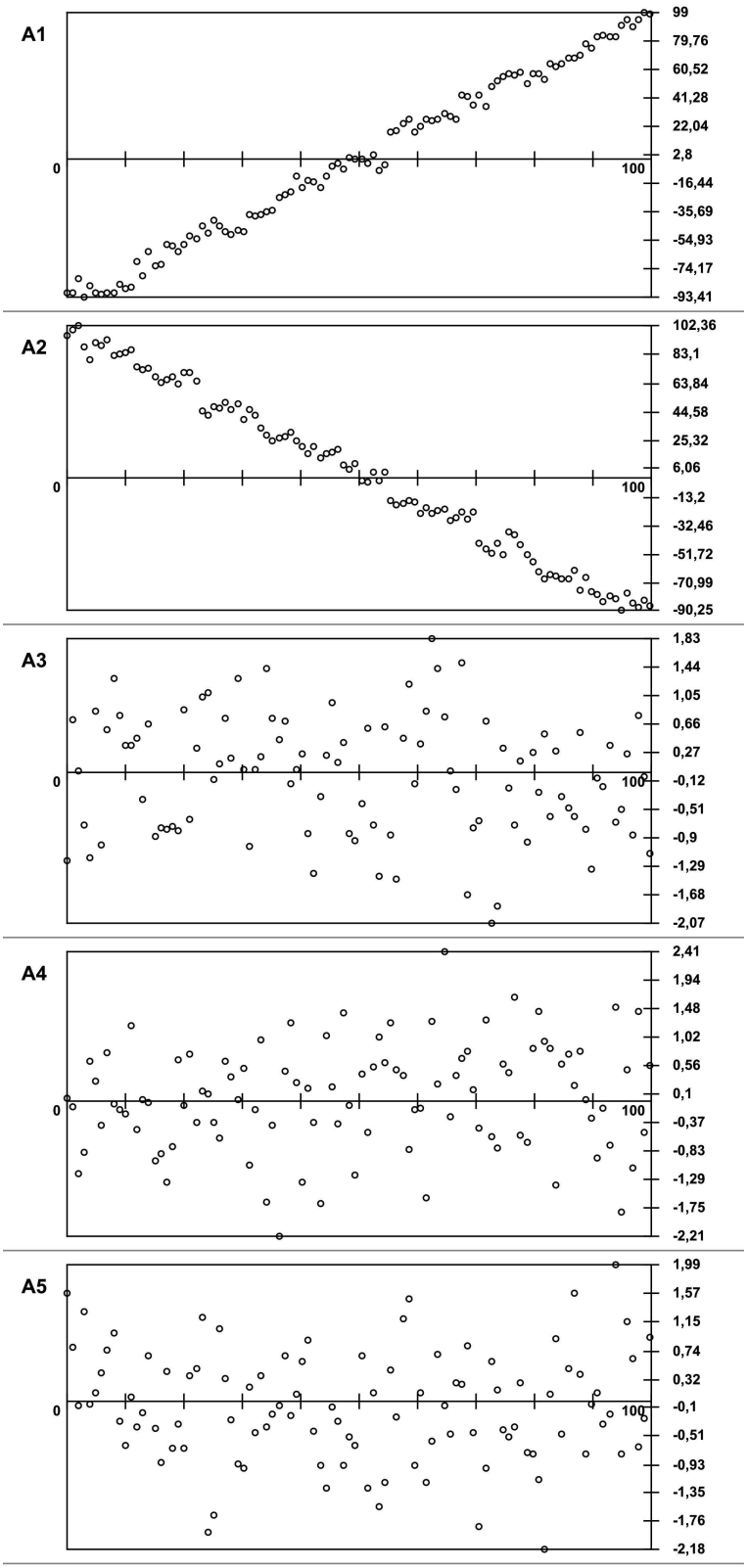

Dataset: dRandom
Model: LR
A1
0,05 0,04 0,03 0,02 0,01 -0 -0,01 -0,02 -0,03 -0,04 -0,05
0 100
A2
0,04 0,03 0,02 0,02 0,01 -0 -0,01 -0,02 -0,03 -0,04 -0,05
0 100
A3
0,05 0,04 0,03 0,02 0,01 0 -0,01 -0,02 -0,03 -0,04 -0,05
0 100
A4
3,34 2,68 2,02 1,36 0,7 0,04 -0,63 -1,29 -1,95 -2,61 -3,27
0 100
A5
0,04 0,03 0,03 0,02 0,01 -0 -0,01 -0,02 -0,03 -0,04 -0,05
0 100
Dataset: dRandom
Model: MP
A1
1,09 0,78 0,46 0,14 -0,17 -0,49 -0,8 -1,12 -1,44 -1,75 -2,07
0 100
A2
3,95 3,26 2,57 1,87 1,18 0,49 -0,2 -0,89 -1,58 -2,27 -2,97
0 100
A3
3,59 0,45 -2,68 -5,82 -8,95 -12,09 -15,22 -18,36 -21,49 -24,63 -27,76
0 100
A4
2,41 1,17 -0,06 -1,3 -2,53 -3,77 -5 -6,24 -7,47 -8,71 -9,94
0 100
A5
7,22 5,46 3,7 1,95 0,19 -1,57 -3,33 -5,09 -6,85 -8,61 -10,37
0 100

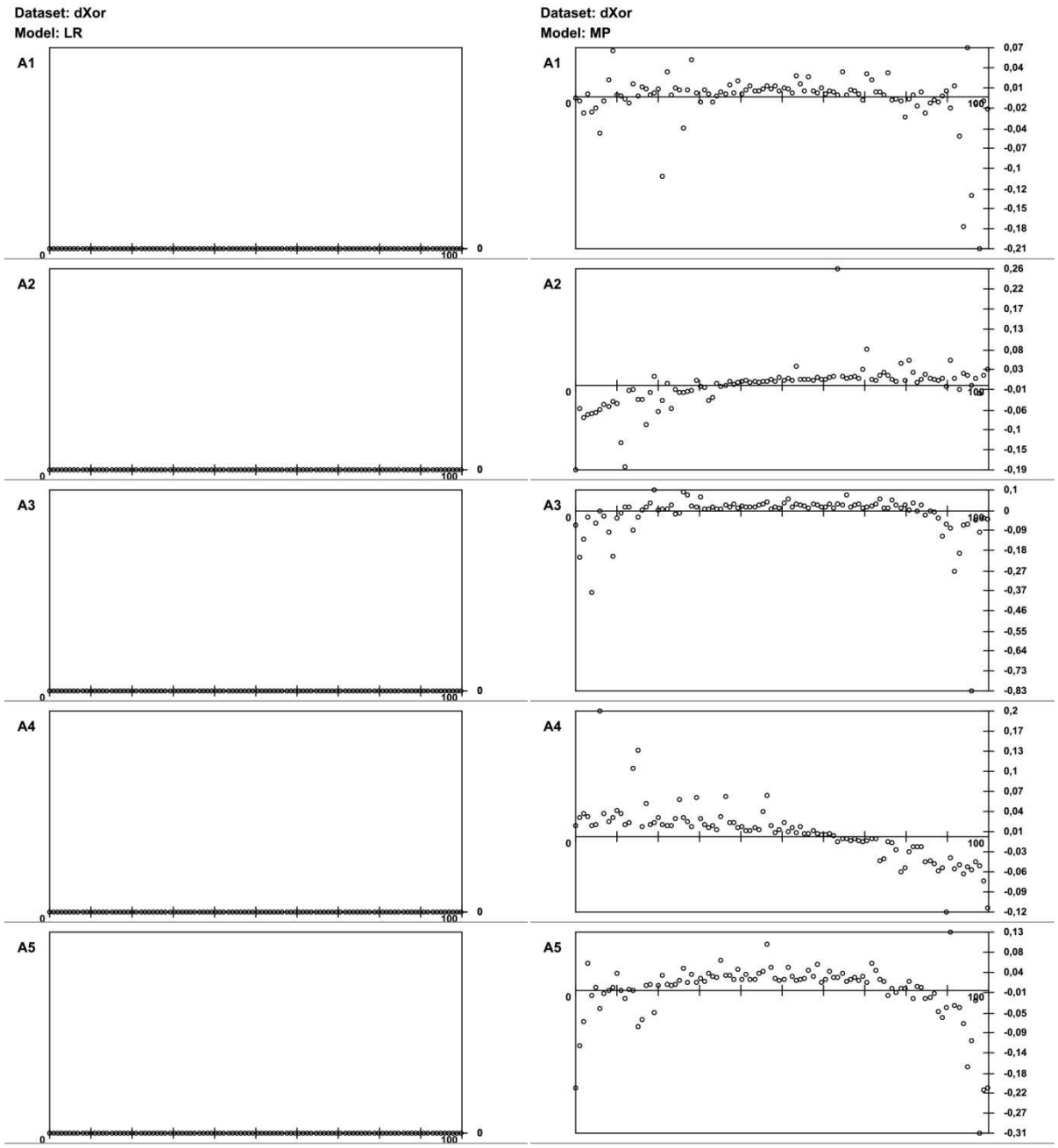

Dataset: dRandom
Model: M5P
A1
11,47
8,26
5,06
1,86
-1,35
-4,55
-7,76
-10,96
-14,17
-17,37
-20,57
0
100
A2
2,54
1,92
1,31
0,7
0,09
-0,53
-1,14
-1,75
-2,36
-2,98
-3,59
0
100
A3
1,5
0,47
-0,56
-1,59
-2,62
-3,65
-4,68
-5,71
-6,74
-7,77
-8,8
0
100
A4
18,17
15,75
13,33
10,91
8,5
6,08
3,66
1,24
-1,18
-3,6
-6,01
0
100
A5
0,14
0,11
0,07
0,04
0
-0,03
-0,06
-0,1
-0,13
-0,17
-0,2
0
100

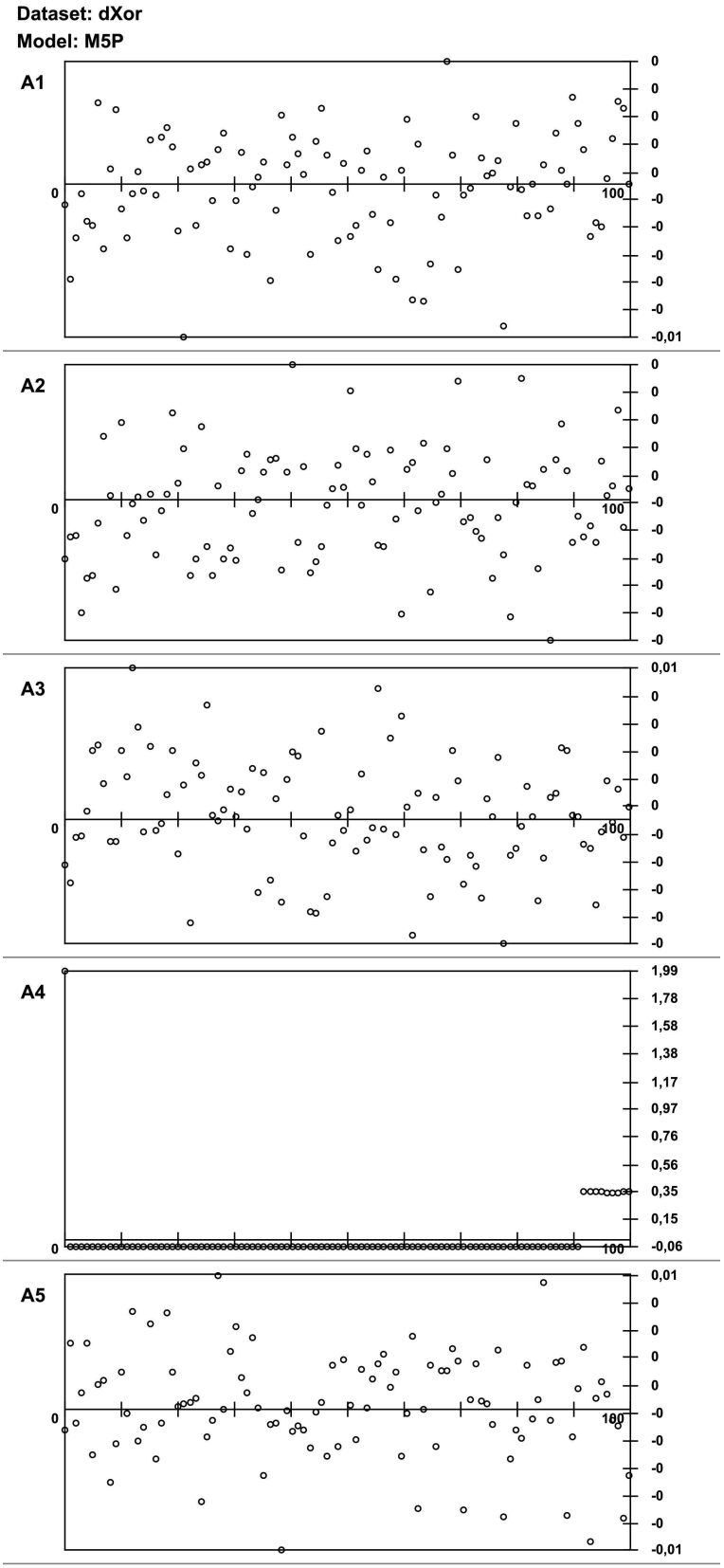

Dataset: dRedundant
Model: LR
A1
99,73
79,96
60,19
40,42
20,65
0,87
-18,9
-38,67
-58,44
-78,21
-97,98
0
100
A2
102
82,41
62,83
43,24
23,66
4,08
-15,51
-35,09
-54,68
-74,26
-93,85
A3
1,81
1,43
1,05
0,67
0,29
-0,09
-0,47
-0,85
-1,23
-1,61
-1,99
A4
1,87
1,5
1,12
0,75
0,38
0
-0,37
-0,75
-1,12
-1,5
-1,87
A5
1,94
1,54
1,14
0,74
0,33
-0,07
-0,47
-0,87
-1,27
-1,67
-2,07

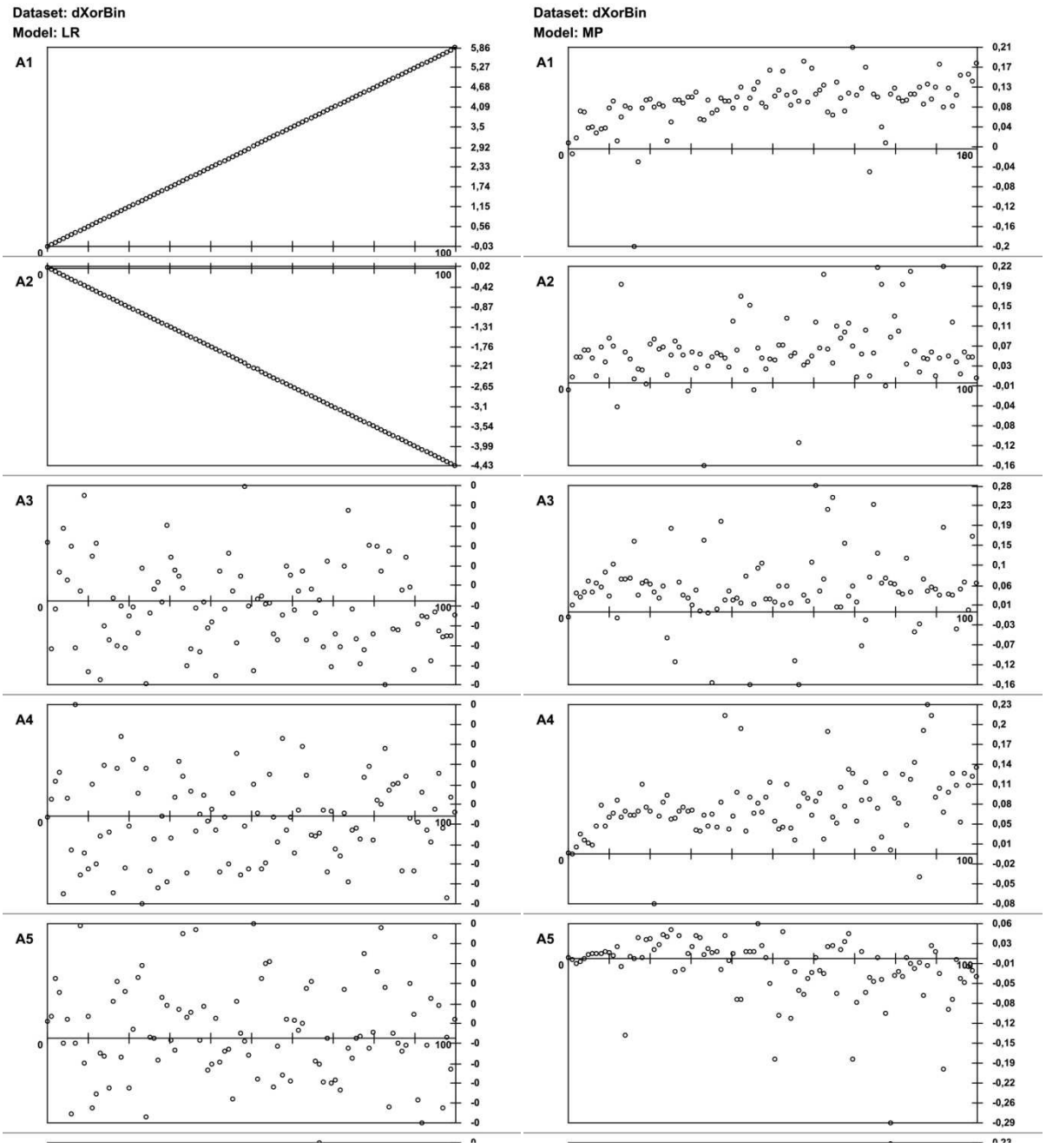

Dataset: dRedundant
Model: MP
A1
99,71
79,82
59,94
40,06
20,18
0,3
-19,58
-39,46
-59,34
-79,23
-99,11
0
100
A2
49,04
39,55
30,06
20,57
11,07
1,58
-7,91
-17,4
-26,89
-36,38
-45,87
A3
54,38
43,87
33,36
22,84
12,33
1,82
-8,69
-19,2
-29,72
-40,23
-50,74
A4
2,92
2,39
1,86
1,33
0,8
0,27
-0,26
-0,79
-1,32
-1,85
-2,38
A5
1,7
1,32
0,95
0,57
0,2
-0,18
-0,55
-0,93
-1,3
-1,68
-2,05

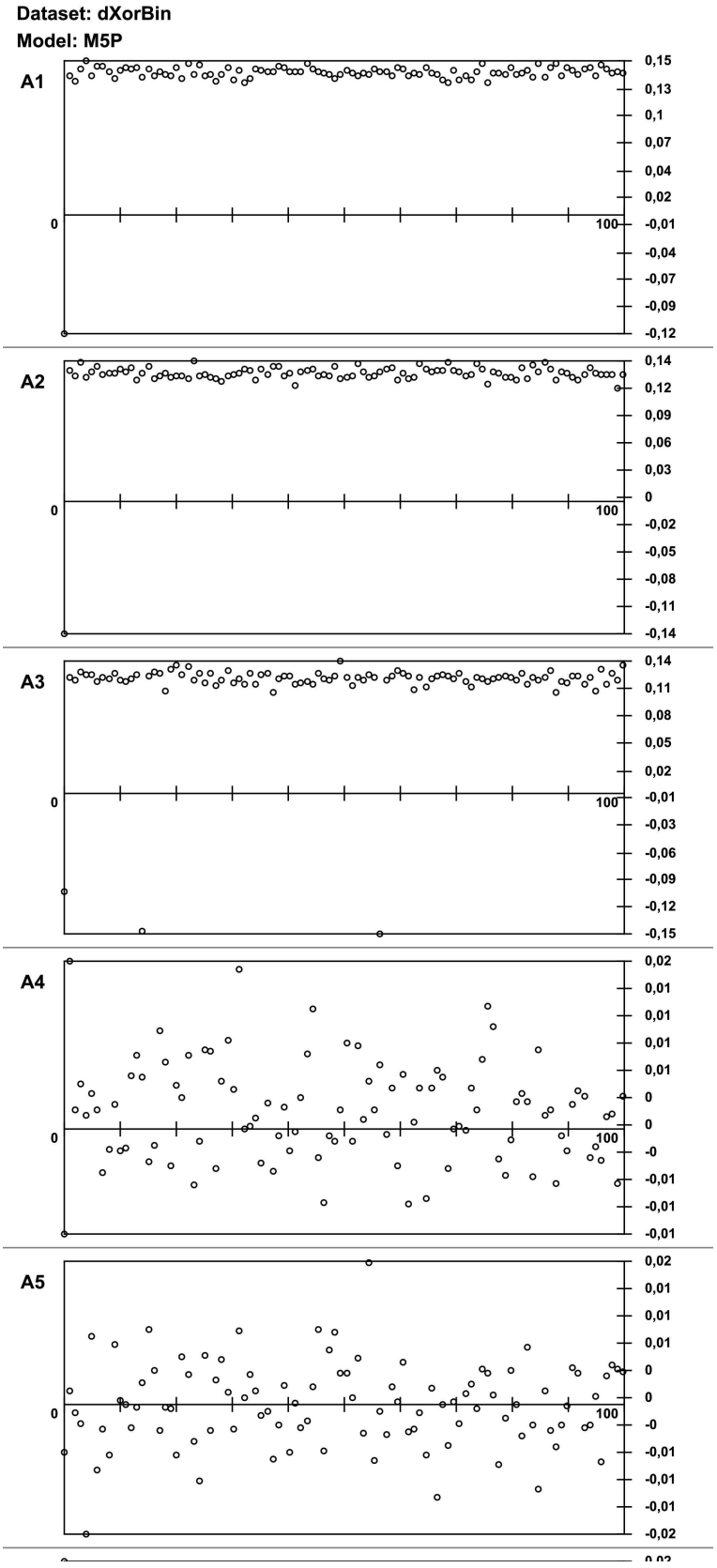
Dataset: dRedundant
Model: M5P




Dataset: dXor
Model: LR
A1
A2
A3
A4
A5
Dataset: dXor
Model: MP
A1
0,07
0,04
0,01
-0,02
-0,04
-0,07
-0,1
-0,12
-0,15
-0,18
-0,21
A2
0,26
0,22
0,17
0,13
0,08
0,03
-0,01
-0,06
-0,1
-0,15
-0,19
A3
0,1
0
-0,09
-0,18
-0,27
-0,37
-0,46
-0,55
-0,64
-0,73
-0,83
A4
0,2
0,17
0,13
0,1
0,07
0,04
0,01
-0,03
-0,06
-0,09
-0,12
A5
0,13
0,08
0,04
-0,01
-0,05
-0,09
-0,14
-0,18
-0,22
-0,27
-0,31
0
100


Dataset: dXor
Model: M5P
A1
A2
A3
A4
A5
0
100
1,99
1,78
1,58
1,38
1,17
0,97
0,76
0,56
0,35
0,15
-0,06
0,01
-0
-0,01


Dataset: dXorBin
Model: LR
A1
A2
A3
A4
A5
Dataset: dXorBin
Model: MP
A1
A2
A3
A4
A5


Dataset: dXorBin
Model: M5P
A1
0,15
0,13
0,1
0,07
0,04
0,02
-0,01
-0,04
-0,07
-0,09
-0,12
0
100
A2
0,14
0,12
0,09
0,06
0,03
0
-0,02
-0,05
-0,08
-0,11
-0,14
0
100
A3
0,14
0,11
0,08
0,05
0,02
-0,01
-0,03
-0,06
-0,09
-0,12
-0,15
0
100
A4
0,02
0,01
0,01
0,01
0,01
0
0
-0
-0,01
-0,01
-0,01
0
100
A5
0,02
0,01
0,01
0,01
0
0
-0
-0,01
-0,01
-0,01
-0,02
0
100

VIRI

[1] E. Štrumbelj, I. Kononenko, "An Efficient Explanation of Individual Classifications using Game Theory," *Journal of Machine Learning Research (to appear in 2010).*

[2] J. Klauser, I. Kononenko, "Explanation of regression decisions by analogy with the explanation in classification.", v: *Zbornik Information Society*, 12.-16. oktober 2009, vol. A, p.46-49

[3] I. Kononenko, *Strojno učenje*, 2. popravljena izdaja, Založba FE in FRI, Ljubljana, 2005.

[4] Developer resources for Java Technology. Dostopno na: http://java.sun.com/

[5] Eclipse, an integrated development environment platform. Dostopno na: http://www.eclipse.org/

[6] Data Mining with Open Source Machine Learning Software in Java. Dostopno na: http://www.cs.waikato.ac.nz/ml/weka/

[7] A java EPS library. Dostopno na: http://jlibeps.sourceforge.net/

[8] Java Statistical Classes. Dostopno na: http://www.jsc.nildram.co.uk/

[9] Attribute-Relation File Format. Dostopno na: http://www.cs.waikato.ac.nz/~ml/weka/arff.html