Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Alojzij Blatnik

Napovedovanje kritičnih dogodkov pri ponudbi računskih virov

diplomsko delo

visokošolski strokovni študijski program prve stopnje računalništvo in informatika

Mentor: prof. dr. Marko Robnik Šikonja

Ljubljana, 2012

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00274/2012
Datum: 11.04.2012

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

Kandidat: **ALOJZIJ BLATNIK**

Naslov: **NAPOVEDOVANJE KRITIČNIH DOGODKOV PRI PONUDBI RAČUNSKIH VIROV**

FORCASTING CRITICAL EVENTS IN PROVISION OF COMPUTATIONAL RESOURCES

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

Za upravnike računske infrastrukture je pomembno, da znajo natančno predvideti porabo posameznih virov in nastop morebitnih ozkih grl in izjemnih dogodkov.
Analizirajte podatke o zasedenosti računskih virov danega podjetja in poskušajte napovedati rabo in izjemne dogodke za določen čas naprej. Uporabite metode za napovedovanje časovnih vrst ter jih med seboj primerjajte. Proučite tudi smiselnost in način uporabe metod za podatkovno rudarjenje.

Mentor:
prof. dr. Marko Robnik Šikonja

Dekan:
prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Alojzij Blatnik, z vpisno številko **63090262**,
sem avtor diplomskega dela z naslovom:

Napovedovanje kritičnih dogodkov pri ponudbi računskih virov

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom prof. dr. Marka Robnik Šikonje,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 5. september 2012 Podpis avtorja:

Zahvaljujem se mentorju prof. dr. Marku Robnik Šikonji za literaturo, strokovne nasvete in podporo pri pisanju diplomskega dela.

Kazalo vsebine

Kazalo slik

Kazalo tabel

Povzetek

Podjetja informacijske tehnologije imajo pogosto veliko število naprav, med katere spadajo strežniki, mrežna oprema, tiskalniki, naprave za zagotavljanje neprekinjenega napajanja itd. Pri velikem številu naprav postane dobro in neprekinjeno delovanje infrastrukture težje obvladljivo, zato je za zgodnje odkrivanje problemov ključnega pomena periodični nadzor naprav. Pridobljene podatke poskušamo analizirati in na njihovi podlagi odkrivati izjemne dogodke. V nalogi so opisane metode, s katerimi avtomatsko ali polavtomatsko odkrivamo anomalije pri nadzoru izbrane opreme, in ustrezna nastavitev njihovih parametrov.

Ključne besede: nadzor opreme, avtomatsko odkrivanje izjem, izjemni dogodki, časovne vrste, ARIMA

Abstract

Information technology companies usually own many servers, network devices, printers, uninterruptible power supplies, etc. A large number of devices is difficult to manage and to assure their uninterrupted service and early problem detection a periodic device monitoring is essential. We analyze the collected data and try to detect anomalies. We describe methods for automatic or semi-automatic detection of anomalies in the functioning of selected devices as well as adequate settings of their parametres.

Keywords: device monitoring, automatic detection of anomalies, anomalies, time series, ARIMA

1 Uvod

Večja podjetja ali računalniško usmerjena podjetja imajo tipično večje število naprav. Pri večjih računalniških omrežjih postane to neobvladljivo in morebitne nepravilnosti niso več trivialne za odkriti. Pri strežnikih na primer ni nujno, da je napačno nastavljeno konfiguracijo mogoče odkriti takoj po vzpostavitvi sistema. Napak se pri večjih računalniških omrežjih slej ko prej nabere še več in v robnih primerih pripeljejo do odpovedi storitve ali vseh odvisnih storitev. Odpovedi storitve se je mogoče v veliki večini izogniti, če je le ta odkrita na podlagi siptomov.

Monitoriranje računalniških omrežij periodično zbira podatke in jih shranjuje v podatkovno bazo. Za odkrivanje problema je potrebno opazovati več parametrov naenkrat. Ti so navadno v korelaciji in ni nujno, da so vsi na isti napravi. Primer: odpoved delovanja omrežne povezave povzroči spremembo v delovanju na več računalnikih.

Diplomska naloga obravnava avtomatsko odkrivanje nepravilnosti v časovni vrsti (periodično pridobljeni podatki). Zaradi kompleksnosti so predstavljene metode, ki omogočajo opazovanje le ene časovne vrste naenkrat.

Diplomska naloga proučuje in primerja naslednje metode: zaznavanje z visokimi in nizkimi mejami, zaznavanje glede na odstopanje v preteklosti, zaznavanje glede na število prekoračitev v časovnem obdobju, ARIMA.

V drugem poglavju opisujemo problem in podatke. Tretje poglavje opisuje delovanje metod in postopkov za odkrivanje anomalij. V četrtem poglavju opisujemo delovanje metod na konkretnih podatkih ter dosežene rezultate posameznih metod. Zaključujemo s komentiranjem rezultatov posameznih metod in ideje za nadaljno delo.

2 Opis problema in podatkov

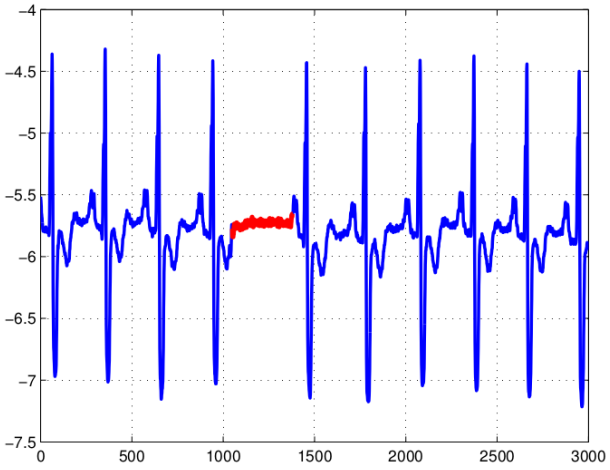
Opišemo problem in podatke, ki se zbirajo pri nadzoru naprav.

2.1 Opis problema

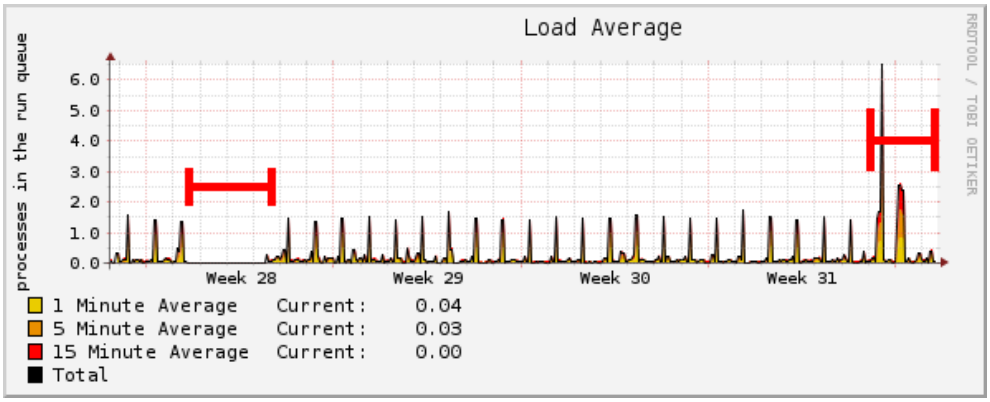
Pri nadzoru velikega števila naprav se za vsako napravo zbira veliko število parametrov, zato postane nadzor nad celotno infrastrukturo težaven. Zaželeno je, da pri nadzoru infrastrukture ne bi bilo potrebno neprestano pregledovanje vseh parametrov. Problem rešujemo z uporabo metod, ki zaznajo nepričakovane podatke in odkrijejo izjemni dogodek. Cilj teh metod je čimbolje označiti začetek in konec izjemnega dogodka. Konec izjemnega dogodka pomeni, da sprememba v sistemu postane normalno stanje ali da se stanje normalizira.

2.1.1 Izjemni dogodki

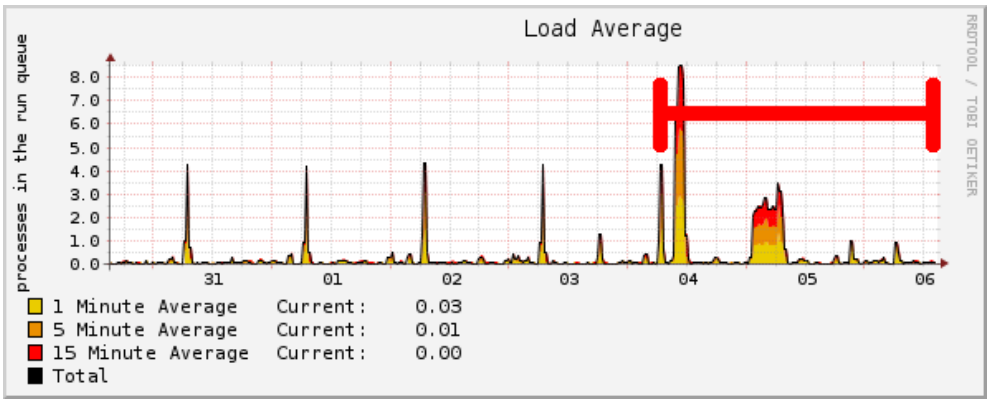
Anomalija ali izjemni dogodek v časovni vrsti je vzorec podatkov, ki statistično ne ustrezajo dosedanjim vzorcem [1]. Postopek odkrivanja izjem je iskanje vzorcev, ki ne ustrezajo pričakovanemu obnašanju [1]. Primeri izjemnih dogodkov so na slikah 1, 2, 3 in 4. Slika 1 prikazuje izjemni dogodek v človeškem EKG. Slika 2 prikazuje vrsto povprečne zasedenosti procesorja na mesečnem območju. Vrsta povprečne zasedenosti procesorja je v začetku nepričakovano nizka, proti koncu je vidno drugačno obnašanje. Slika 3 prikazuje obremenitev na tedenskem intervalu. Slika 4 prikazuje spremembo v zasedenosti diska.



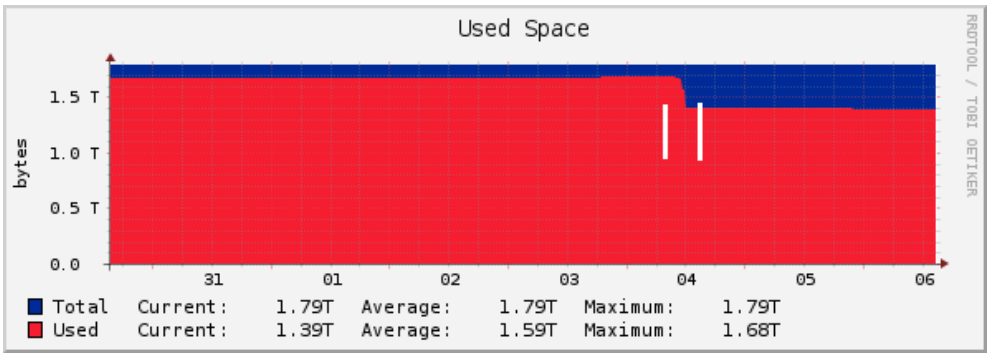
Slika 1: Izjemni dogodek v človeškem EKG [1].



Slika 2: Vrsta povprečne zasedenosti procesorja.



Slika 3: Vrsta povprečne zasedenosti procesorja.



Slika 4: Sprememba v zasedenosti diska.

2.2 Opis podatkov

Pri nadzoru opreme v računalnišem omrežju se periodično zbirajo podatki s posameznih naprav. Periodično zbiranje podatkov definira množico časovnih vrst, ki so lahko med seboj odvisne. V našem primeru smo zaradi kompleksnosti opazovali le eno časovno vrsto naenkrat.

2.2.1 Zbiranje podatkov

Večino podatkov smo zbirali prek protokola SNMP (Simple Network Management Protocol), nekaj podatkov smo zbirali tudi po drugih poteh.

Primeri podatkov, ki smo jih zbirali prek protokola SNMP:

- zasedenost diskov,
- obremenjenost procesorjev,
- dolžino vrste (load average),
- število prijavljenih uporabnikov,
- zasedenost mrežnih vmesnikov, ...

Primeri podatkov, ki smo jih zbrali po drugih poteh:

- ping: merjenje zakasnitve med strežnikom ter ciljno napravo,
- merjenje pasovne širine: meritve števca na omrežnem vmesniku ob dveh zaporednih časovnih mejnikih,
- število prenešenih paketov pri IPTV vmesniku: analiza statistike na modemu.

Primer pridobivanja podatka prek protokola SNMP:

```
$ snmpwalk -On -v2c -c public localhost . | head
.1.3.6.1.2.1.1.1.0 = STRING: "Linux dark-acer 3.2.0-23-generic #36-Ubuntu
SMP Tue Apr 10 20:39:51 UTC 2012 x86_64"
.1.3.6.1.2.1.1.2.0 = OID: .1.3.6.1.4.1.8072.3.2.10
.1.3.6.1.2.1.1.3.0 = Timeticks: (4729207) 13:08:12.07
.1.3.6.1.2.1.1.4.0 = STRING: "Me <me@example.org>"
.1.3.6.1.2.1.1.5.0 = STRING: "dark-acer"
.1.3.6.1.2.1.1.6.0 = STRING: "Sitting on the Dock of the Bay"
```

```
.1.3.6.1.2.1.1.7.0 = INTEGER: 72
.1.3.6.1.2.1.1.8.0 = Timeticks: (17) 0:00:00.17
.1.3.6.1.2.1.1.9.1.2.1 = OID: .1.3.6.1.6.3.10.3.1.1
.1.3.6.1.2.1.1.9.1.2.2 = OID: .1.3.6.1.6.3.11.3.1.1
```

Dolžina vrste v 15-minutnem časovnem oknu:

```
$ uptime
 00:55:45 up 13:13,  1 user,  load average: 1.59, 1.60, 1.69
$ snmpwalk -On -v2c -c public localhost .1.3.6.1.4.1.2021.10.1.3.3
.1.3.6.1.4.1.2021.10.1.3.3 = STRING: "1.69"
```

Za podatke, ki jih ni možno pridobiti prek protokola SNMP in jih je možno pridobiti na drugačen način, smo napisali skripte. Primer odčitavanja števca prenešenih podatkov na IPTV vmesniku:

```
$ cat mc.py
#!/usr/bin/python2

import urllib
import html2text
import sys, time, os

try:
    page = os.popen('ssh root@192.168.2.1 "wget http://user:xxx@192.168.1
.1/webconfig/status/traffic_stats.html -O -" 2>/dev/null').read()
    parsedPage = html2text.html2text(page).split()
    failedWAN = parsedPage[51]
    succount = failedWAN.split("/")[0]
    errcount = failedWAN.split("/")[1]
    out="succount:"+succount+" errcount:"+errcount
    sys.stdout.write(out)
except:
    sys.stdout.write("succount:3 errcount:3")
```

Primer klica:

```
$ python mc.py
succount:40170268 errcount:310
```

Za periodično zbiranje podatkov smo uporabili ogrodje Cacti [6]. Ogrodje Cacti periodično (v našem primeru vsakih 5 minut) zbere podatke in jih shrani v RRD (Round Robin Database) podatkovno bazo. Podatki v round

robin podatkovni bazi se s časom redčijo, s tem baza ohranja fiksno velikost. Ta lastnost je dobra, saj na "pozabljenem" sistemu ne more zasesti diska, pa tudi zadnji, pogosto bolj relevantni podatki so bolj pogosto vzorčeni. Ogrodje Cacti podpira arhitekturo vtičnikov, kjer se lahko preko vtičov pridobi ali spreminja vmesne podatke. Da smo metode lažje testirali, smo uporabili vtič, prek katerega smo shranjevali podatke v podatkovno bazo in s tem ohranili vso zgodovino.

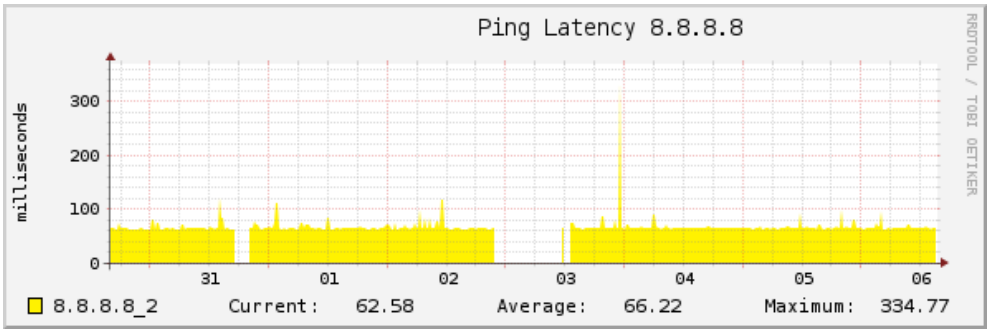
Primer uporabljenih podatkov, shranjenih v podatkovni bazi:

```
mysql> describe all_data2;
+---------------+-------------+------+-----+---------+----------------+
| Field         | Type        | Null | Key | Default | Extra          |
+---------------+-------------+------+-----+---------+----------------+
| id            | bigint(20)  | NO   | PRI | NULL    | auto_increment |
| rrdfilename   | varchar(900)| YES  |     | NULL    |                |
| local_data_id | int(11)     | YES  |     | NULL    |                |
| ds_name       | varchar(100)| YES  |     | NULL    |                |
| time          | bigint(20)  | YES  |     | NULL    |                |
| value         | varchar(100)| YES  |     | NULL    |                |
+---------------+-------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)

mysql> select id, time, value
    -> from all_data2
    -> where rrdfilename="/var/www/cacti/rra/datastore_load_15min_277.rrd"
    -> and ds_name="load_15min" and time>1342315917 and time<1342316917
    -> order by time;

+---------+------------+-------+
| id      | time       | value |
+---------+------------+-------+
| 1700605 | 1342316217 | 0.25  |
| 1700669 | 1342316517 | 0.17  |
| 1700733 | 1342316817 | 0.11  |
+---------+------------+-------+
3 rows in set (0.83 sec)
```

2.2.2 Manjkajoči podatki

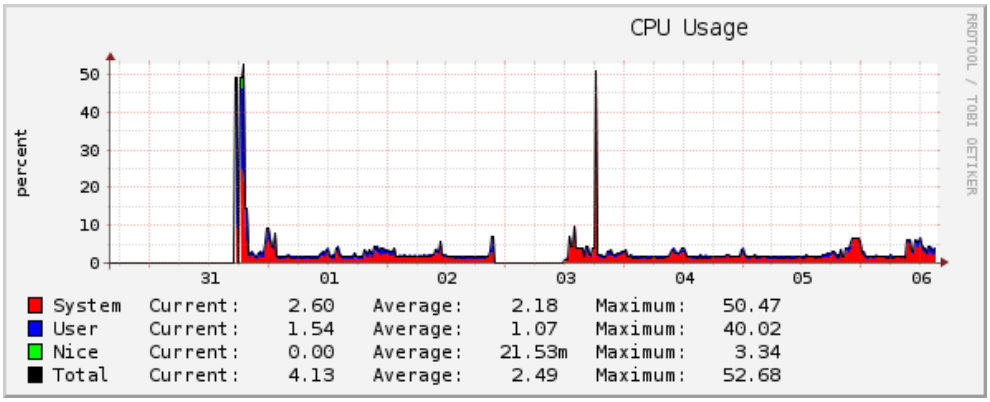
Zaradi različnih vzrokov je zajem podatkov lahko tudi neuspešen, npr.:

- merjenje zakasnitve ping: odpoved povezave do ciljne naprave ali odpoved strežnika za nadzor (slika 5),



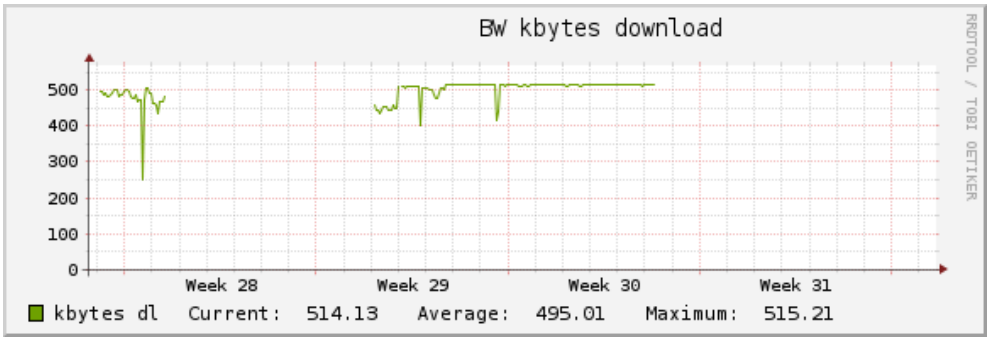
Slika 5: Manjkajoči podatki v ping grafu.

- merjenje zasedenosti procesorja: odpoved ciljne naprave, mrežne povezave ali odpoved strežnika za nadzor (slika 6),



Slika 6: Manjkajoči podatki v CPU grafu.

- merjenje pasovne širine: odpoved ciljnega strežnika, usmerjevalnika za odčitavanje števca, mrežne povezave ali odpoved strežnika za nadzor (slika 7).



Slika 7: Manjkajoči podatki v grafu za merjenje pasovne širine.

Obstajajo metode, ki manjkajoče podatke zapolnijo z napovedanimi vrednostmi in algoritmi, ki delujejo z nepopolnimi podatki. V nalogi smo manjkajoče podatke obravnavali pri vsaki metodi posebaj, saj je njihova obdelava precej različna za vsak algoritem.

3 Opis metod za napovedovanje časovnih vrst

Opišemo delovanje posameznih metod, ki smo jih uporabili v nalogi.

3.1 Zaznavanje z visokimi in nizkimi mejami

Algoritem zaznavanja z visokimi in nizkimi mejami je enostavna metoda, ki javi izjemo, če je vrednost večja ali manjša od navedenih pragov.
Metoda dobro deluje pri diskretnih vrednostih.
Primer: stanje ventilatorja. 0 - ni podatka, 1 - deluje, 2 - ne deluje. Smiselno je nastaviti tako nizko kot visoko mejo na vrednost 1, saj je to normalno stanje.

Pri številskih vrednostih nam metoda pomaga le, kadar vrednost preseže minimalno ali maksimalno mejo, ne more pa odkriti nestandardih vzorcev.
Primer: zasedenost diska. Visoko vrednost lahko nastavimo na 80 %. Algoritem bo sicer zaznal, ko bo začelo primanjkovati prostora, vendar ne prej, kot je nastavljena meja.

3.2 Zaznavanje glede na pretekla odstopanja

Zaznavanje glede na odstopanje v preteklosti sporoči izjemo, kadar je sprememba večja ali manjša od navedenih minimalnih ali maksimalnih odklonov. Algoritem opazuje okno n točk, kjer je n sodo število, in ga razdeli na dva dela. Za oba dela okna izračuna povprečje ter javi izjemo, kadar je razlika večja ali manjša od navedenih pragov.
Večje kot je okno, na daljšem intervalu zaznavamo spremembo povprečja. Zaznavanje glede na pretekla odstopanja je, kot zaznavanje z visokimi in nizkimi mejami, neinteligentna metoda in zahteva veliko pozornosti pri nastavljanju mej. Algoritem deluje dobro, če je tok podatkov umirjen, kot nepričakovan dogodek pa štejemo nenadno spremembo.
Primer: sprememba v zasedenosti diska. Smiselno je uporabiti metodo skupaj z zaznavanjem z visokimi in nizkimi mejami, saj ob natančni nastavitvi parametrov opišemo dovoljen odklon ter maksimalne vrednosti.

3.3 Zaznavanje glede na število prekoračitev v časovnem obdobju

Izjema zaradi število prekoračitev v časovnem obdobju se sproži, če je v nekem časovnem obdobju število prekoračitev nekega algoritma ali parametra večje ali manjše od nastavljenega praga.

Primer: zasedenost prometa na omrežnem vmesniku vsak dan razen sobote in nedelje doseže vsaj 90 % obremenitev. V tem primeru je smiselno nastaviti zaznavanje z visokimi in nizkimi mejami ter kot minimalno in maksimalno dovoljeno število sprožitev na 5, kjer je časovno okno 1 teden. Kadar algoritem sproži izjemo, ugotovimo, da vmesnik ni dosegel pričakovane obremenitve.

3.4 ARIMA

Model ARIMA je splošen model za napovedovanje časovnih vrst [2, 3]. Pri odkrivanju izjemnih dogodkov nam pomaga tako, da napove n naslednjih vrednosti in za vsako napovedano vrednost poda standardni odklon. Napovedane vrednosti primerjamo z dejanskimi vrednostmi, kjer za minimalno ter maksimalno vrednost upoštevamo določeno število standardnih odklonov. Izjema se sproži, če je dejanska vrednost zunaj izračunane minimalne ali maksimalne vrednosti.
Za model je pomembno, da so vhodni podatki stacionarni (angl. stationary). V primeru, da vhodni podatki niso stacionarni (vsebujejo trend), se v integracijskem delu izvede transformacija, da podatki postanejo stacionarni. Metoda ARIMA(p,d,q) (Autoregressive Integrated Moving Average) je sestavljena iz avtoregresijskega dela AR(p), integracijskega dela I(d) in drsečega povprečja MA(q) [9].

Avtoregresijski del

Avtoregresijski del AR(p) je osnovan na ideji, da je trenutna vrednost x_t izračunljiva grede na zadnje p vrednosti $x_{t-1}, x_{t-2}, ..., x_{t-p}$, kjer p predstavlja število potrebnih korakov za napoved trenutne vrednosti.
Avtoregresijski model je definiran po naslednji formuli:

$$x_t = \theta_1 x_{t-1} + \theta_2 x_{t-2} + ... + \theta_p x_{t-p} + w_t \tag{1}$$

kjer je x_t stacionaren ali rezultat integracijskega dela npr. I(1), $\theta_1, \theta_2, ..., \theta_p$ konstante ($\theta_p \neq 0$) ter w_t beli šum s povprečjem 0 in standardnem odklonom σ_w^2 razporejen po Gausovi porazdelitvi [9].

Integracijski del

Integracijski del I(d) z diferencialom d iz časovne vrste odstranjuje trend. Število d pomeni število razlik pri pretorbi časovne vrste v stacionarno obliko [2].

Drseče povprečje

Za razliko od avtoregresijskega dela, ki upošteva zamike predhodnih vrednostih, drseče povprečje upošteva zamike v napakah [2].

Drseče povprečje MA(q) je definirano kot:

$$x_t = w_t + \theta_1 w_{t-1} + \theta_2 w_{t-2} + ... + \theta_q w_{t-q} \tag{2}$$

kjer q pomeni število zamikov v drsečem povprečju, $\theta_1, \theta_2, ..., \theta_q$ so parametri, w_t pa je beli šum s povprečjem 0 ter standardnim odklonom σ_w^2 razporejen po Gausovi porazdelitvi.

Določanje parametrov

Pri določanju parametrov p in q uporabimo Box-Jenkins metodo, ki najde najboljše ujemanje na preteklih vrednostih. Parametera skoraj nikoli nimata vrednosti večje od 2. Do vrednosti 2 uporabimo približek iz tabele 1, ki določa vrednosti parametra p in q glede na obliko ACF (Autocorrelation function) ter PACF (Partial autocorrelation function) [5].

p = 1	ACF eksponentno pada; PACF ima konico na zakasnitvi 1, brez ujemanja v preostanku
p = 2	ACF oblika sinusa ali več eksponentnih oblik; PACF konica na zakasnitvi 1 in 2, brez ujemanja v preostanku
q = 1	ACF konica na zakasnitvi 1, brez ujemanja v preostanku; PACF pada exponentno
q = 2	ACF konica na zakasnitvi 1 in 2, brez ujemanja v preostanku; PACF sinusna oblika ali več eksponentnih oblik
p = 1 q = 1	ACF eksponentno pada po zakasnitvi 1; PACF eksponentno pada po zakasnitvi 1

Tabela 1: Določanje parametra p in q na podlagi ACF ter PACF.

Napovedovanje

ARIMA model se pri napovedovanju obnaša kot funkcija. Pri napovedovanju bo pri $d = 0$ konstanten trend, $d = 1$ linearen trend in $d = 2$ kvadraten trend. Napovedovanje se izvede po obrazcu [9]:

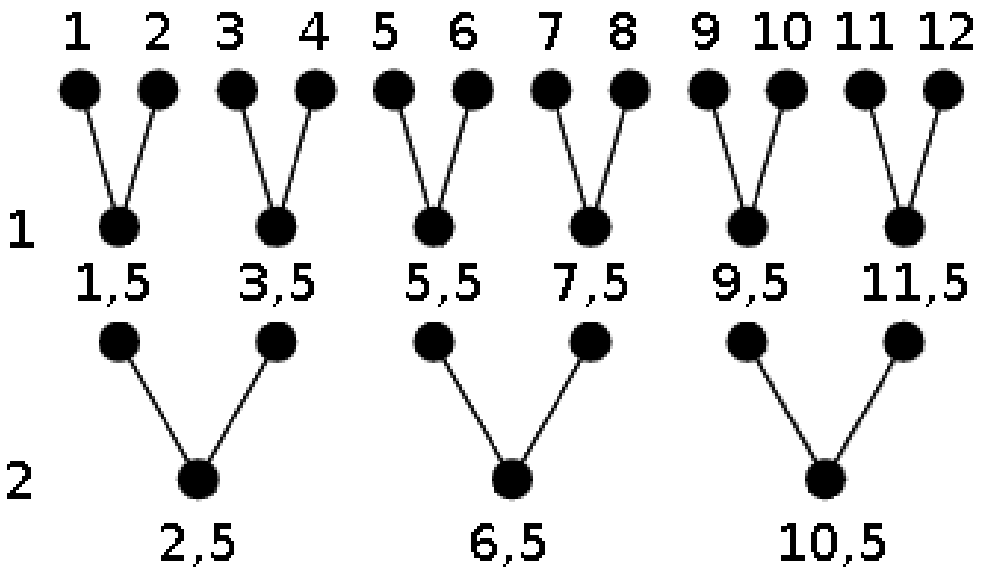
$$x_{n+m} - \tilde{x}_{n+m} = \sum_{j=0}^{m-1} \psi_j w_{n+m-j} \tag{3}$$

4 Opis delovanja metod na konkretnih podatkih

Opišemo rezultate posamezne metode na realnih podatkih ter komentiramo rezultate. Izjemne dogodke smo odkrivali na dnevnem ter urnem intervalu. Priprava podatkov in implementacija numeričnih metod je bila izvedena v programskem jeziku Python. Pri metodi ARIMA smo uporabili implementacijo v sistemu R [7], za povezavo programskega jezika Python in R smo uporabili python modul rpy2 [8]. Zavedati se moramo, da vsak model drugače odkriva izjeme ter da na numerične metode vplivamo le s parametri. Zaradi tega bomo pri vsaki metodi obravnavali časovne vrste, za katere je ta metoda namenjena. Proti koncu bomo komentirali rezultate posameznih metod.

4.1 Zgoščanje podatkov

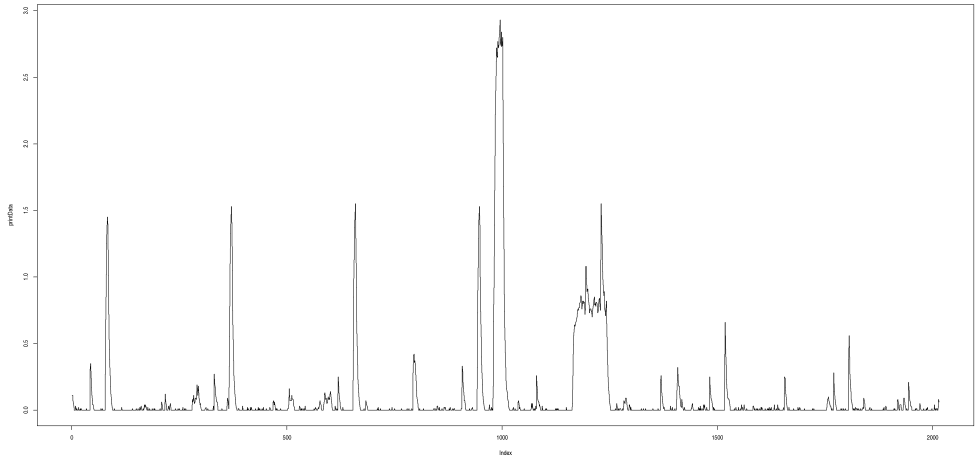
Pri dnevnem intervalu smo zaradi hitrejšega procesiranja ter manjšega števila konic zgostili podatke. Zgoščali smo po metodi binarnega drevesa, kjer se na vsakem koraku izračuna povprečje. Prikaz dveh iteracij zgoščanja prikazuje slika 8.



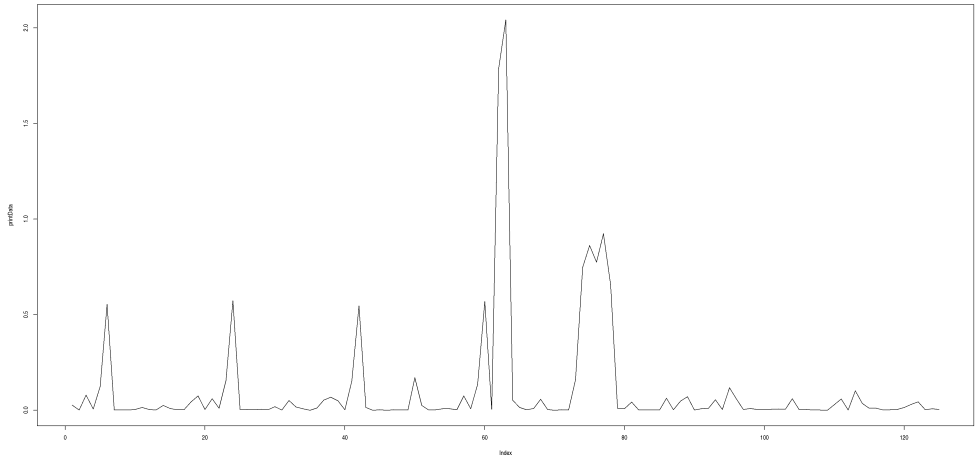
Slika 8: Zgoščanje podatkov.

Pri dnevnem intervalu smo podatke zgostili tako, da je imel interval 18 vrednosti (4 iteracije zgoščanja). Pri urni napovedi nismo zgoščali podatkov

(imamo 12 podatkov na uro zaradi 5 minutnega vzorčenja). Primerjava zgoščenih in nezgoščenih podatkov za teden je vidna na slikah 9 in 10.



Slika 9: Primer nezgoščenega okna.



Slika 10: Primer zgoščenega okna.

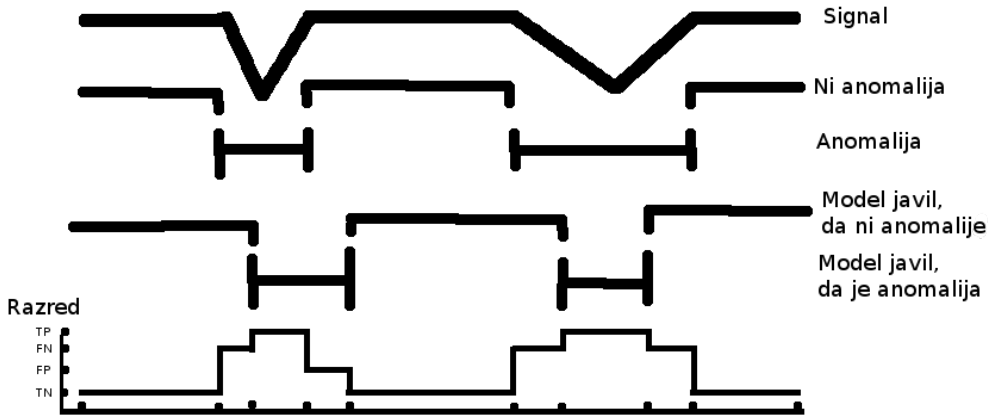
4.2 Vrednotenje rezultatov

Kot rezultat vsakega modela smo dobili seznam intervalov, pri katerih je model zaznal izjemo.
Vsak interval je bil uvrščen v enega izmed razredov:

- TP (True Positives) - interval je izjema in model je pravilno zaznal, da je izjema,
- TN (True Negatives) - interval ni izjema in model je pravilno zaznal, da ni izjema,
- FP (False Positives) - interval ni izjema, vendar je model zmotno zaznal kot izjema,
- FN (False Negatives) - interval je izjema, vendar je model ni zaznal.

Rezultate smo vrednotili:

- z deležem pokritja in zgradili tabelo napačnih klasifikacij. Razreda FN in FP smo cenovno utežili (razred FN ima 100 krat višjo ceno). Slika 11 prikazuje primer vrednotenja z deležem pokritja.



Slika 11: Primer vrednotenja z deležem pokritja.

  Iz tabele napačnih klasifikacij smo izračunali senzitivnost, specifičnost in klasifikacijsko točnost [10].

- Z delem zaznanih izjem: merili smo, ali je bila dejanska izjema zaznana, in ugotovili, ali obstaja presek med izjemami, ki jih je odkril model, in

dejanskimi izjemami.
Rezultat dobimo v odstotkih deleža zaznanih izjem:

$$delež\ zaznanih\ izjem = \frac{število\ zaznanih\ izjem}{vseh\ izjem} * 100 \qquad (4)$$

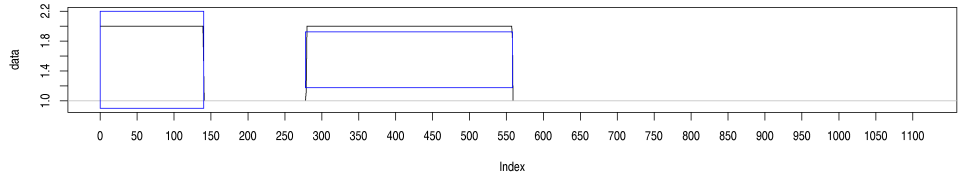
- Z razmerjem dejanskih in zaznanih izjem: model lahko doseže dobro senzitivnost in visok delež zaznanih izjem s tem, da javlja preveč izjem. Takšne primere (npr. model javi, da je kar celotno območje izjema) smo zaznavali s specifičnostjo.

$$razmerje = \frac{število\ javljenih\ izjem}{vseh\ izjem} \qquad (5)$$

4.3 Zaznavanje z visokimi in nizkimi mejami

Metoda zaznavanja z visokimi in nizkimi mejami odkriva prekoračitve v časovni vrsti. Aktivna je toliko časa, da se stanje spet normalizira. V našem primeru pri odkrivanju na dnevnem ali urnem intervalu zaradi narave delovanja ni bilo razlik. Rezultati so odvisni le od nastavite visoke in nizke meje. Manjkajoče vrednosti ta metoda ignorira in zato reagira z zamikom.
Pri nastavitvi parametrov smo pozorni na vrsto vhodnih podatkov:

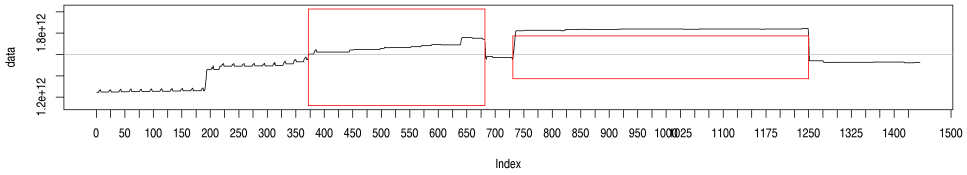
- diskretne vrednosti:
  Metoda pri diskretnih vrednostih deluje idealno (slika 12) in doseže 100% klasifikacijsko točnost. Na sliki 12 dejanska vrednost (modra barva) popolnoma prekriva rezultat modela (rdeča barva).



Slika 12: Odkrivanje v diskretni časovni vrsti. Simuliran podatek: 1 - ok, 2 - napaka

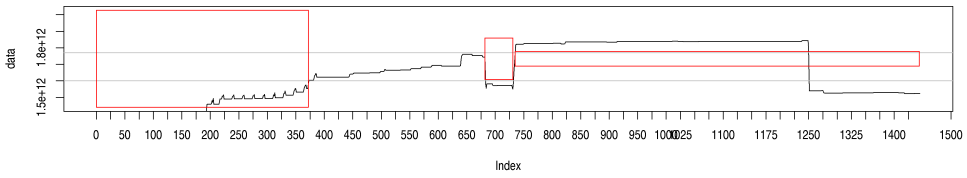
- umirjena številska časovna vrsta (ne vsebuje izrazitih periodičnih konic, ki obsegajo večino zaloge vrednosti):
  V umirjeni številski časovni vrsti metodo uporabimo tako, da nastavimo

zgornjo mejo na nek odstotek maksimalne vrednosti ali na določeno fiksno vrednost, če maksimalne vrednosti ne poznamo. Slika 13 prikazuje graf zasedenosti diska, kjer je nastavljena samo zgornja meja. Samo zgornjo mejo je smiselno nastaviti, kadar je meja dovolj nizka.



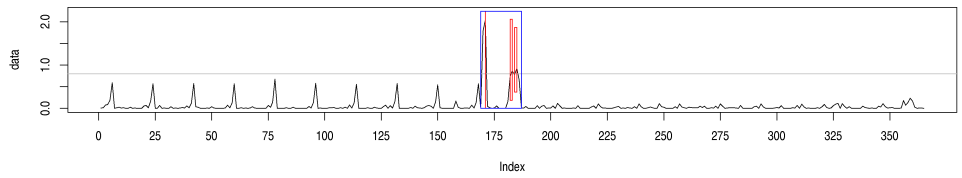
Slika 13: Zasedenost diska, nastavljen zgornji prag.

Ker je metoda aktivna, dokler se stanje ne normalizira, jo "ustavimo" z nastavitvijo pragov tako, da je dejanska vrednost med nizkim in visokim pragom. Slika 14 prikazuje spremembo nastavitve pragov za območje od 372 do 682. Tako nastavljeni pragi zaznajo, manj ali bolj zaseden disk.



Slika 14: Zasedenost diska, nastavljen zgornji in spodnji prag.

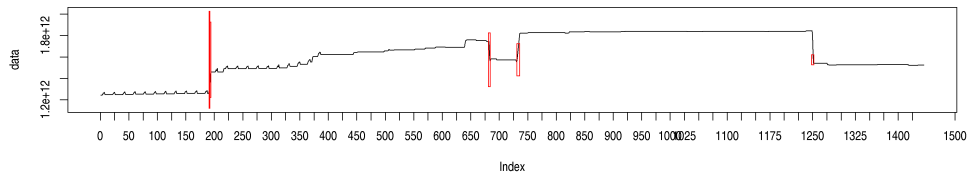
- periodično nihajoče številske vrednosti:
  Pri periodičnem nihanju smo prisiljeni mejo dvigniti, da nam izjeme ne javlja vsak cikel, zato izjemo zaznamo kasneje. Slika 15 prikazuje mejo dvignjeno nad periodične amplitude (vrednost 0.8) in zaznani izjemi, ki sta prekoračili mejo.



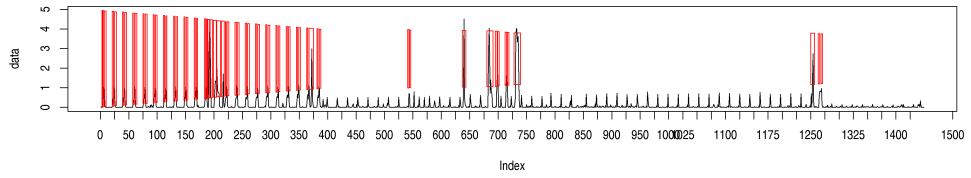
Slika 15: Vrsta povprečne zasedenosti procesorja.

4.4 Zaznavanje glede na pretekla odstopanja

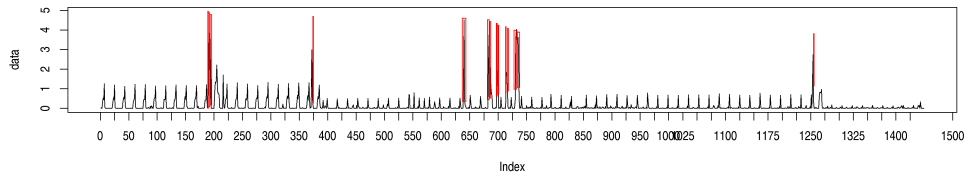
Metoda zaznavanja glede na pretekla odstopanja odkriva spremembe v časovni vrsti. Manjkajoče vrednosti ignorira. V našem primeru pri odkrivanju na dnevnem ali urnem intervalu zaradi narave delovanja ni bilo razlik. Metoda deluje dobro, če ni veliko sprememb v časovni vrsti, sicer moramo dobro nastaviti prage, pri kateri želimo, da metoda javi izjemo. Slika 16 prikazuje spremembe v zasedenosti diska, kjer ni potrebno zelo natančno nastaviti pragov, saj so spremembe redke in zelo očitne. Slika 17 prikazuje preobčutljivo nastavljen prag, zato metoda javi preveč izjem. Slika 18 prikazuje bolje nastavljen prag.



Slika 16: Spremembe v zasedenosti diska.



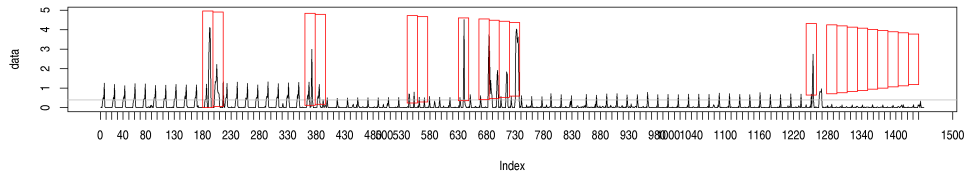
Slika 17: Preobčutljivo nastavljen prag.



Slika 18: Bolje nastavljen prag.

4.5 Zaznavanje glede na število prekoračitev v časovnem obdobju

Metoda zaznavanja glede na število prekoračitev v časovnem intervalu odkrije izjemo, če je prekoračitev manj ali več od navedenega praga v določenem opazovanem oknu. Manjkajoče vrednosti metoda ignorira. Če želimo odkrivati na dnevnem ali urnem intervalu, moramo ustrezno nastaviti velikost opazovanega okna. Na sliki 19 pričakujemo, da bo enkrat na dan prekoračena meja 0.4. V prvem delu je prekoračitev preveč, proti koncu se je stanje spremenilo in metoda na vsakem koraku javi premalo prekoračitev.



Slika 19: Število prekoračitev nastavljeno na 1.

4.6 ARIMA

Pri modelu ARIMA smo uporabili implementacijo iz sistema R, ki zgradi več modelov z različnimi parametri in uporabi model, ki da najboljše rezultate. Manjkajoče podatke je obvravnaval v izbranem modelu.

Klic funkcije:

```
"data.fit <- auto.arima(window(data,start=c(1,%d), end=c(1,%d)))"
                                % (self.arima_start,self.arima_end)
```

Napoved:

```
"data.pred <- predict(data.fit, n.ahead=%d)" % self.predict_ahead
```

Rezultat funkcije predict je vektor napovedi ter vektor standardnih odklonov.

4.6.1 Parametri

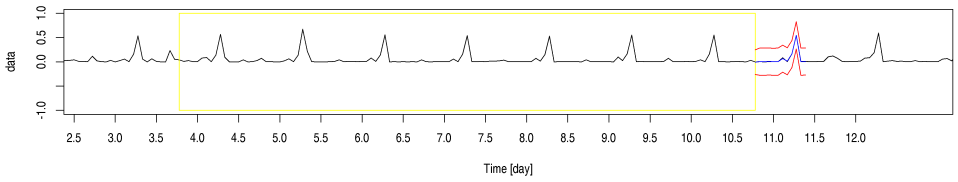
Ker smo uporabili implementacijo, ki zgradi več modelov in uporabi model z najboljšimi rezultati, ni bilo potrebno nastavljati parametrov p, d, q. Odločiti se moramo, pri kolikšnem odklonu želimo reagirati. Visoko in nizko mejo smo izračunali z množenjem standardnih odklonov:

```
self.se_high = list(ro.r("data.pred$pred+%d*data.pred$se"
                                        % self.stdOffsets))
self.se_low = list(ro.r("data.pred$pred-%d*data.pred$se"
                                        % self.stdOffsets))
```

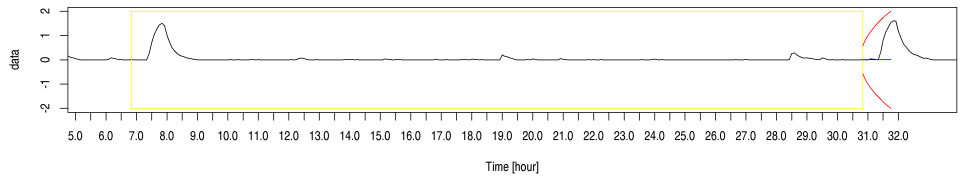
Pri preizkušanju smo ugotovili, da je 10 standardnih odklonov dovolj, da model ne zazna manj očitnih odklonov v umirjenem podatkovnem toku, še vedno pa zazna izrazite odklone.

4.6.2 Podatkovno okno

Pri urnem intervalu je bila učna faza 24 ur, 25. uro pa je model napovedoval. Pri dnevnem intervalu je bila učna faza modela 7 dni, 8. pa je model napovedoval. Slika 20 prikazuje podatkovno okno na dnevnem intervalu (učna faza 7 dni, napoveduje 8. dan). Slika 21 prikazuje podatkovno okno na urnem intervalu (učna faza 24 ur, napoveduje 25. uro).



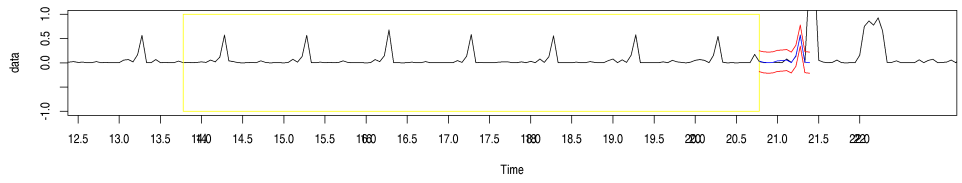
Slika 20: Primer podatkovnega okna na dnevnem intervalu.



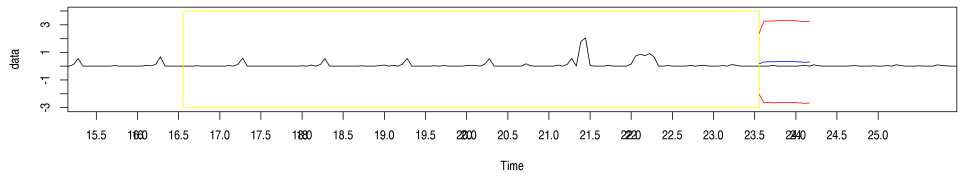
Slika 21: Primer podatkovnega okna na urnem intervalu.

4.6.3 Rezultati

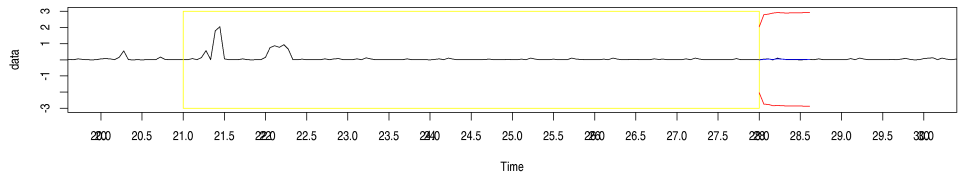
Metoda ARIMA odkriva spremembe v časovni vrsti (slika 22, rdeča oznaka). Po ugotovljeni izjemi standardni odklon izrazito naraste (slika 23), kar za dolžino časovnega okna onemogoči odkrivanje izjem te velikosti (slika 24). Ko v podatkovnem oknu ni več izjeme, je model znova občutljiv na takšne spremembe (slika 25). Model spregleda tudi zmanjšanje amplitude.



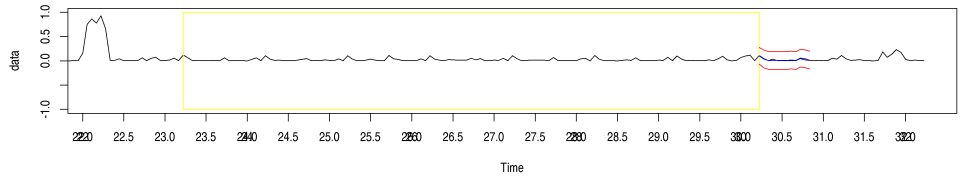
Slika 22: V umirjenem toku ARIMA odkrije izjemo.



Slika 23: Po izjemi standardni odklon naraste.



Slika 24: Povečan standardni odklon vpliva na zaznavanje, dokler je izjema prisotna v podatkovnem oknu.



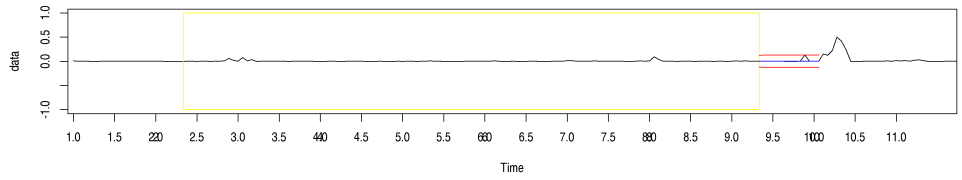
Slika 25: Ponovno normalizirano stanje.

Za vrednotenje rezultatov smo na treh različnih časovnih vrstah istega tipa (vrsta povprečne zasedenosti procesorja) ročno označili izjeme, ki bi jih metoda morala zaznati, in nato izračunali uspešnost zaznavanja. Za primerjavo rezultatov smo modele gradili z različnimi odmiki od podatkovnih oken. Večje kot je, manj modelov se zgradi in več točk klasificira posamezen model. Manjše kot je, natančneje lahko metoda zazna začetek izjeme, vendar moramo pogosteje graditi modele, zato ima večjo časovno zahtevnost.

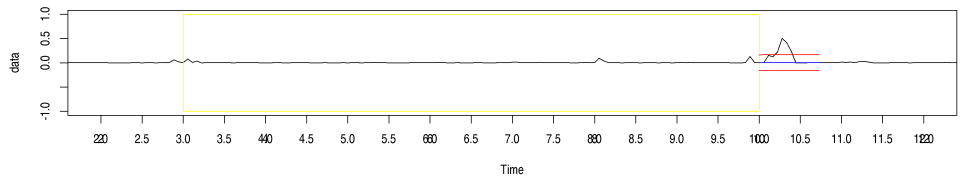
Tabele 2, 3 in 4 v dodatku prikazujejo primerjavo glede na velikost od podatkovnih oken. Opazimo, da imajo modeli slabo senzitivnost in posledično visoko ceno (FN napaka je 100-krat dražja kot FP). To se zgodi zato, ker je metoda označila ožji pas ter pustila območja, ki so enako velika ali manjša od prve izjeme v podatkovnem oknu. Najbolj uporaben podatek pri vrednotenju je delež odkritih izjem. Iz tabel 2, 3 in 4 opazimo, da imajo največji odstotek odkritih izjem modeli pri velikosti oken do 7 korakov odmika.

Pri odmikih okrog 13 in 14 opazimo, da se senzitivnost in delež odkritih izjem povečata. To se zgodi, ker ob različnih odmikih modeli napovedujejo različno daleč. Model lahko pri nekem odmiku ravno zgreši izjemo (slika 26) in jo v naslednji iteraciji odkrije zelo pozno (slika 27). Pri malce večjem

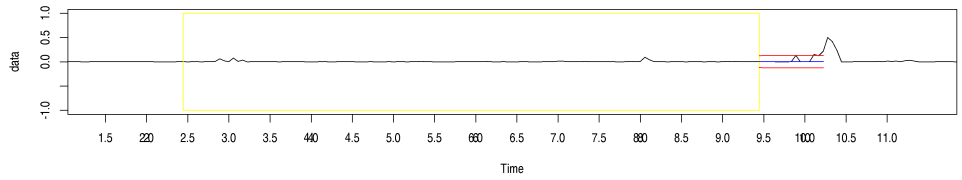
odmiku lahko model izjemo odkrije zelo zgodaj (slika 28). Ta efekt izničimo tako, da modele gradimo dovolj pogosto.



Slika 26: Odmik 12, model ravno zgreši izjemo.

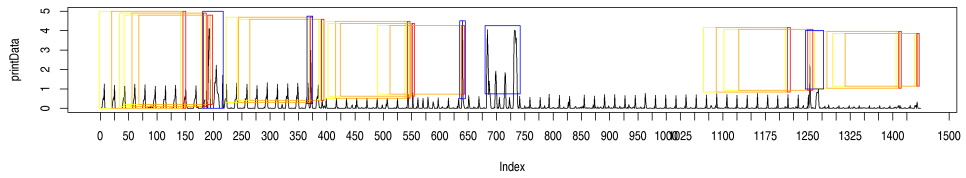


Slika 27: Odmik 12, model izjemo odkrije zelo pozno.

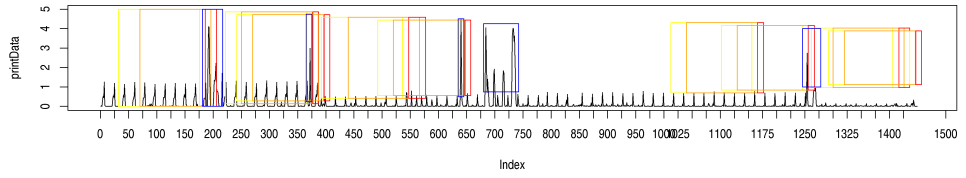


Slika 28: Odmik 13, model izjemo odkrije zelo zgodaj.

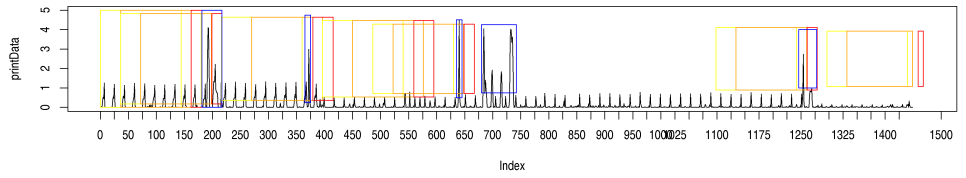
Slike 29, 30 in 31 prikazujejo rezultate iz tabele 2 pri odmikih 4, 10 in 18. Slike 32, 33 in 34 prikazujejo rezultate iz tabele 3, slike 35, 36 in 37 pa prikazujejo rezultate iz tabele 4.
Na slikah je z modro barvo označena dejanska izjema, z rdečo je odkrita izjema, rumena in oranžna označujeta učno fazo metode. Iz slik ugotovimo, da pri večjih odmikih prihaja do večjih zamikov pri odkritju izjem, vendar model pri odmiku 18 še vedno odkrije večino izjem, le z enodnevnim zamikom.



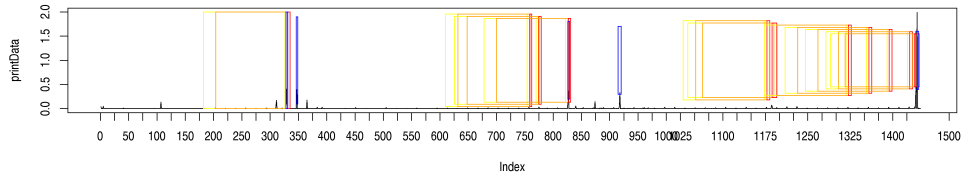
Slika 29: Rezultati pri odmiku 4 za primer 1.



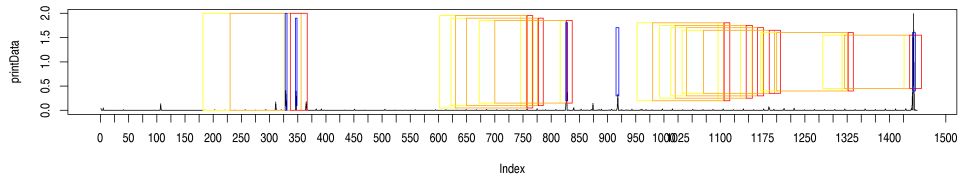
Slika 30: Rezultati pri odmiku 10 za primer 1.



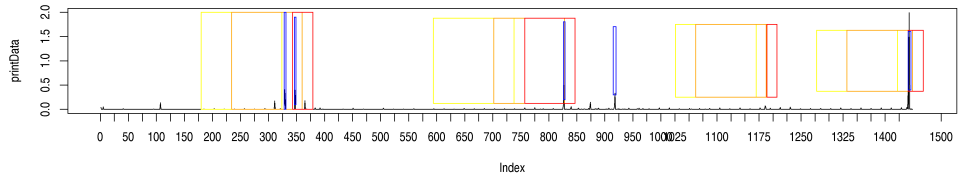
Slika 31: Rezultati pri odmiku 18 za primer 1.



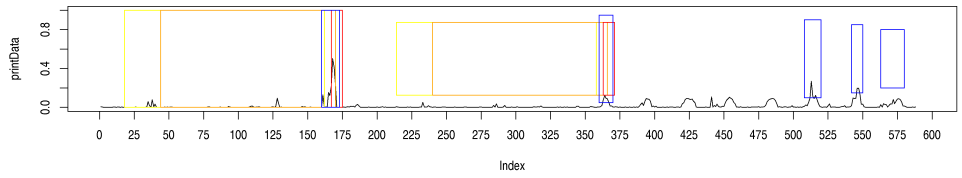
Slika 32: Rezultati pri odmiku 4 za primer 2.



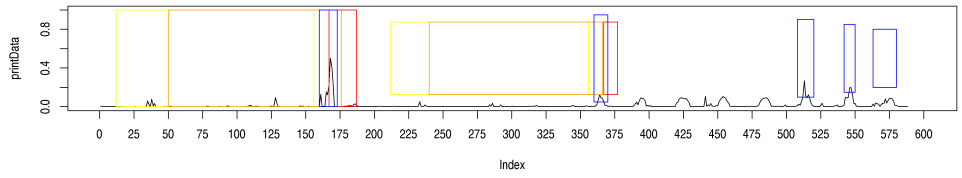
Slika 33: Rezultati pri odmiku 10 za primer 2.



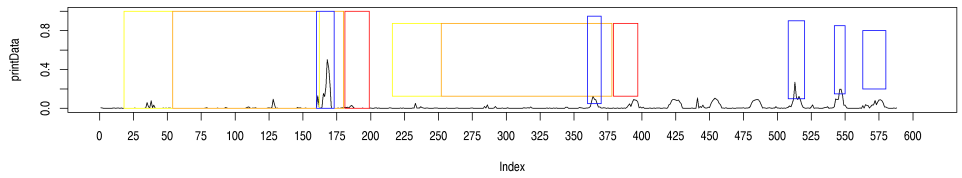
Slika 34: Rezultati pri odmiku 18 za primer 2.



Slika 35: Rezultati pri odmiku 4 za primer 3.



Slika 36: Rezultati pri odmiku 10 za primer 3.



Slika 37: Rezultati pri odmiku 18 za primer 3.

5 Zaključek

V nalogi smo analizirali delovanje treh preprostih numeričnih metod za zaznavanje izjem: zaznavanje z visokimi in nizkimi mejami, zaznavanje glede na pretekla odstopanja in zaznavanje glede na število prekoračitev v časovnem obdobju. Uporabili smo tudi metodo ARIMA. Izjeme smo odkrivali na urnem in dnevnem intervalu. Za odkrivanje na urnem intervalu smo uporabili nezgoščene podatke (12 vrednosti na uro), za odkrivanje na dnevnem intervalu pa zgoščene podatke (18 vrednosti na dan). Rezultati preprostih metod so odvisni izključno od vhodnih parametrov. Ker vsaka metoda odkriva izjeme z drugačnimi cilji, medsebojna primerjava ni smiselna. Uporabili smo implementacijo metode ARIMA iz sistema R, ki je zgradila več modelov in uporabila tistega, ki daje najboljše rezultate. Skušali smo ugotoviti, pri kakšnem odmiku od podatkovnega okna metoda odkrije največ izjem. Na treh časovnih vrstah smo ročno označili izjeme in analizirali delovanje metode ARIMA z različnimi odmiki. Ugotovili smo, da pri manjšem odmiku hitreje odkrijemo anomalijo, vendar ta pristop zahteva večjo časovno zahtevnost. Pri dnevnih napovedih smo ugotovili, da je smiselen odmik manjši od 7, saj odmik 7 pomeni že 9 ur zakasnitve od najhitrejše možnega odkritja (odmik 1). Odmikov na urnem intervalu nismo testirali.

Nadaljnje delo je smiselno usmeriti k:

- drugačnim vrstam obdelave podatkov in novim predstavitvam podatkov;
- odkrivanju izjem z drugačnimi pristopi. Pri metodi ARIMA smo opazovali le, ali je dejanska vrednost odstopala od standardnega odklona. Lahko bi uporabili tudi drugačen pristop in odkrili drugačne izjeme. Smiselno je preučiti tudi delovanje drugih metod, na primer metode strojnega učenja v podatkovnem toku;
- opazovanju več atributov hkrati in s tem razširitvi prostora, kjer je možno odkriti izjeme.

Literatura

[1] (2.9.2012) Varun Chandola, Arindam Banerjee and Vipin Kumar, Anomaly Detection: A Survey, poglavje 1. Dostopno na: http://www.cs.umn.edu/tech_reports_upload/tr2007/07-017.pdf

[2] (2.9.2012) Parts of ARIMA model and fitting. Dostopno na: http://www.forecastingsolutions.com/arima.html

[3] (2.9.2012) Basic applications of ARIMA model. Dostopno na: http://www.duke.edu/ rnau/411arim.htm

[4] (2.9.2012) ARIMA Modelling of Time Series. Dostopno na: http://stat.ethz.ch/R-manual/R-patched/library/stats/html/arima.html

[5] (2.9.2012) How To Identify Patterns in Time Series Data: Time Series Analysis. Dostopno na: http://www.statsoft.com/textbook/time-series-analysis

[6] (2.9.2012) Spletna stran ogrodja Cacti. Dostopno na: http://cacti.net

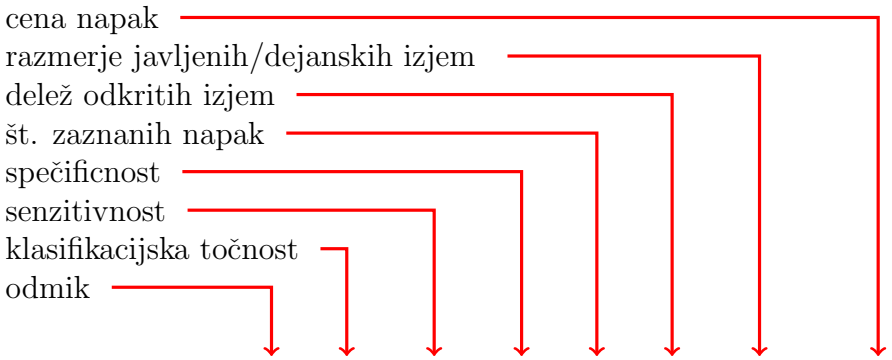
[7] (2.9.2012) Spletna stran sistema R. Dostopno na: http://www.r-project.org

[8] (2.9.2012) Spletna stran modula rpy2. Dostopno na: http://rpy.sourceforge.net

[9] Robert H. Shumway, David S. Stoffer. Time series analysis and its applications, third edition, str. 83 - 85, 116. Založba Springer, 2011.

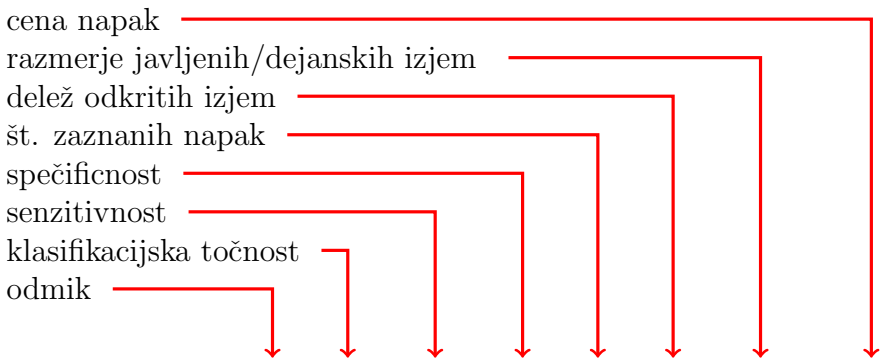
[10] Igor Kononenko, Marko Robnik Šikonja. Inteligentni sistemi, str. 55 - 68. Založba FE in FRI, 2010.

6 Dodatek



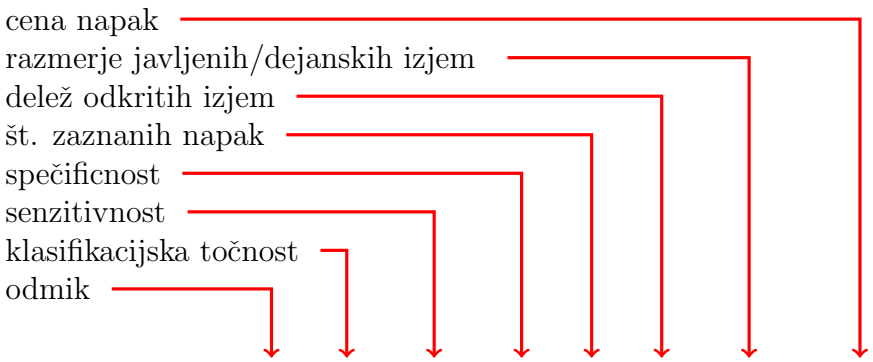
3	0.89	0.09	0.98	9	0.80	1.80	13628.00
4	0.90	0.20	0.98	12	0.80	2.40	12030.00
5	0.88	0.15	0.97	10	0.80	2.00	12743.00
6	0.88	0.11	0.97	8	0.80	1.60	13345.00
7	0.87	0.21	0.95	9	0.80	1.80	11969.00
8	0.86	0.21	0.94	10	0.60	2.00	11977.00
9	0.88	0.27	0.95	8	0.60	1.60	10968.00
10	0.85	0.21	0.93	9	0.40	1.80	11992.00
11	0.87	0.15	0.96	7	0.60	1.40	12757.00
12	0.87	0.18	0.95	5	0.60	1.00	12370.00
13	0.85	0.19	0.93	8	0.60	1.60	12290.00
14	0.86	0.29	0.92	6	0.40	1.20	10703.00
15	0.86	0.28	0.93	6	0.40	1.20	10896.00
16	0.84	0.39	0.89	6	0.60	1.20	9345.00
17	0.87	0.28	0.94	6	0.40	1.20	10877.00
18	0.84	0.23	0.92	7	0.40	1.40	11610.00

Tabela 2: Rezultati glede na ročno označene izjeme, primer 1.



3	0.98	0.58	0.98	10	0.60	2.00	828.00
4	0.96	0.37	0.97	11	0.40	2.20	1250.00
5	0.95	0.37	0.95	11	0.40	2.20	1267.00
6	0.97	0.26	0.98	6	0.20	1.20	1434.00
7	0.94	0.16	0.95	8	0.40	1.60	1677.00
8	0.93	0.42	0.93	9	0.40	1.80	1197.00
9	0.95	0.42	0.96	6	0.40	1.20	1163.00
10	0.91	0.53	0.91	10	0.60	2.00	1023.00
11	0.91	0.16	0.92	6	0.20	1.20	1718.00
12	0.93	0.00	0.94	7	0.00	1.40	1989.00
13	0.92	0.58	0.93	6	0.60	1.20	902.00
14	0.90	0.63	0.90	8	0.80	1.60	837.00
15	0.90	0.16	0.91	7	0.20	1.40	1735.00
16	0.90	0.58	0.90	6	0.60	1.20	943.00
17	0.88	0.26	0.88	8	0.40	1.60	1566.00
18	0.90	0.58	0.90	4	0.60	0.80	941.00

Tabela 3: Rezultati glede na ročno označene izjeme, primer 2.



3	0.91	0.17	1.00	2	0.40	0.40	5002.00
4	0.91	0.22	0.99	2	0.40	0.40	4703.00
5	0.91	0.23	0.99	2	0.40	0.40	4606.00
6	0.89	0.12	0.98	2	0.40	0.40	5311.00
7	0.89	0.15	0.98	2	0.40	0.40	5112.00
8	0.89	0.15	0.97	2	0.40	0.40	5115.00
9	0.87	0.02	0.97	2	0.20	0.40	5917.00
10	0.88	0.15	0.96	2	0.40	0.40	5121.00
11	0.87	0.05	0.96	2	0.40	0.40	5719.00
12	0.87	0.05	0.96	2	0.20	0.40	5721.00
13	0.86	0.12	0.94	2	0.20	0.40	5332.00
14	0.86	0.15	0.94	2	0.40	0.40	5133.00
15	0.86	0.07	0.95	2	0.40	0.40	5626.00
16	0.85	0.05	0.95	2	0.20	0.40	5729.00
17	0.86	0.08	0.95	2	0.20	0.40	5529.00
18	0.84	0.00	0.93	2	0.00	0.40	6036.00

Tabela 4: Rezultati glede na ročno označene izjeme, primer 3.