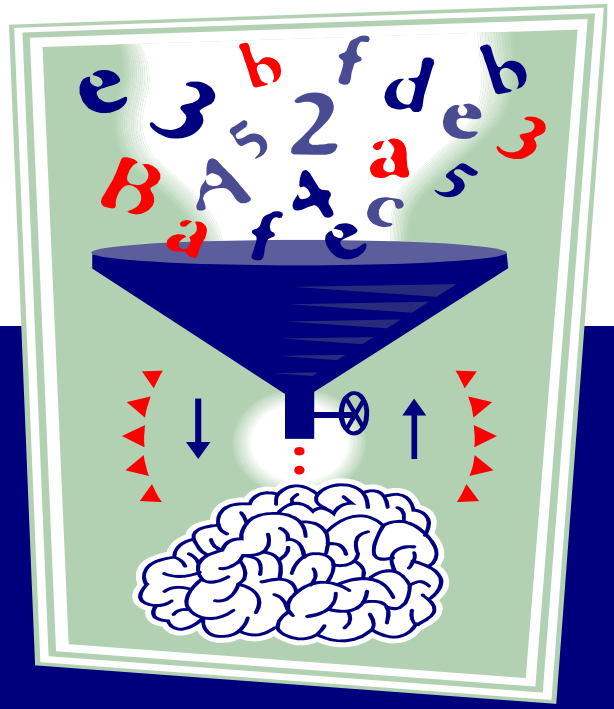


Priprava podatkov za odkrivanje zakonitosti

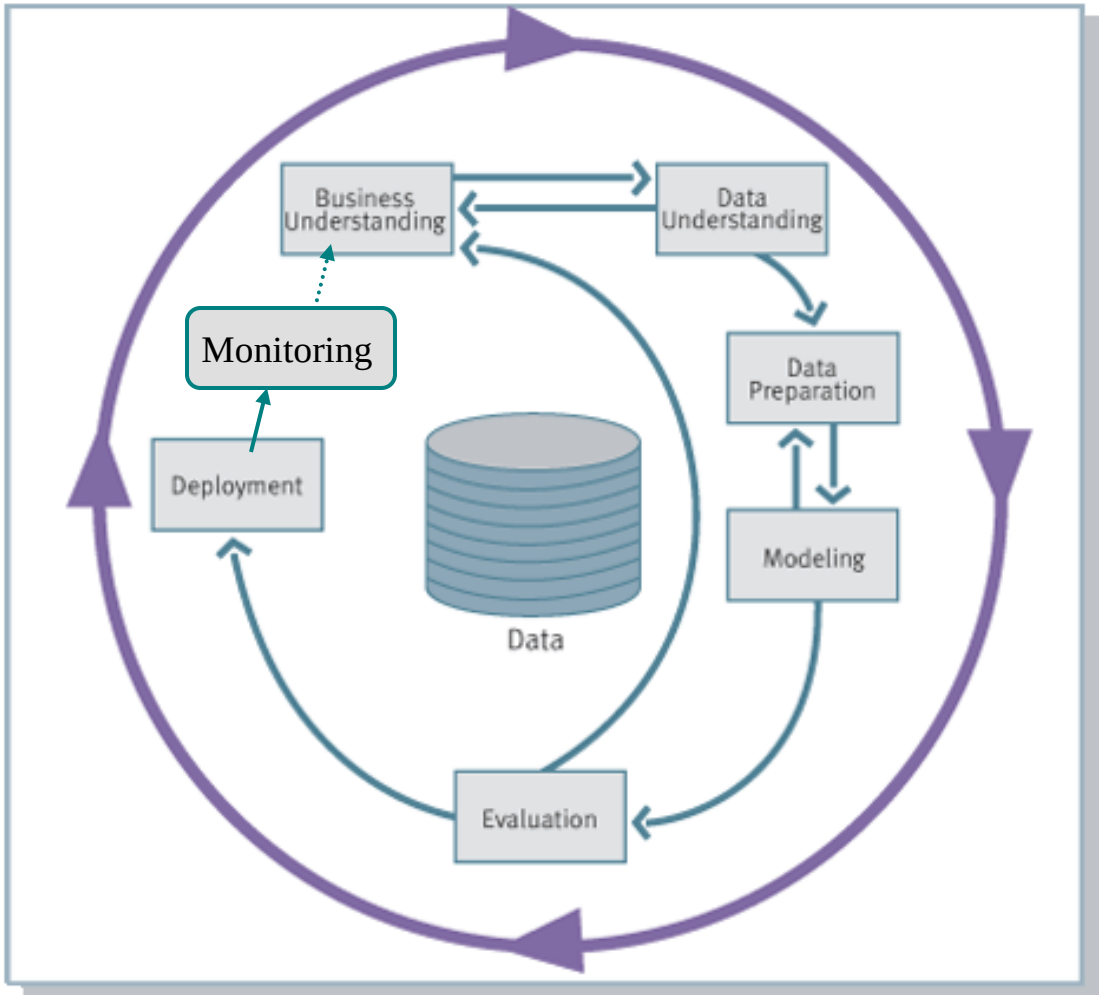
Branko Kavšek
branko.kavsek@upr.si

Osnove strojnega učenja in podatkovnega rudarjenja

Kratko kazalo

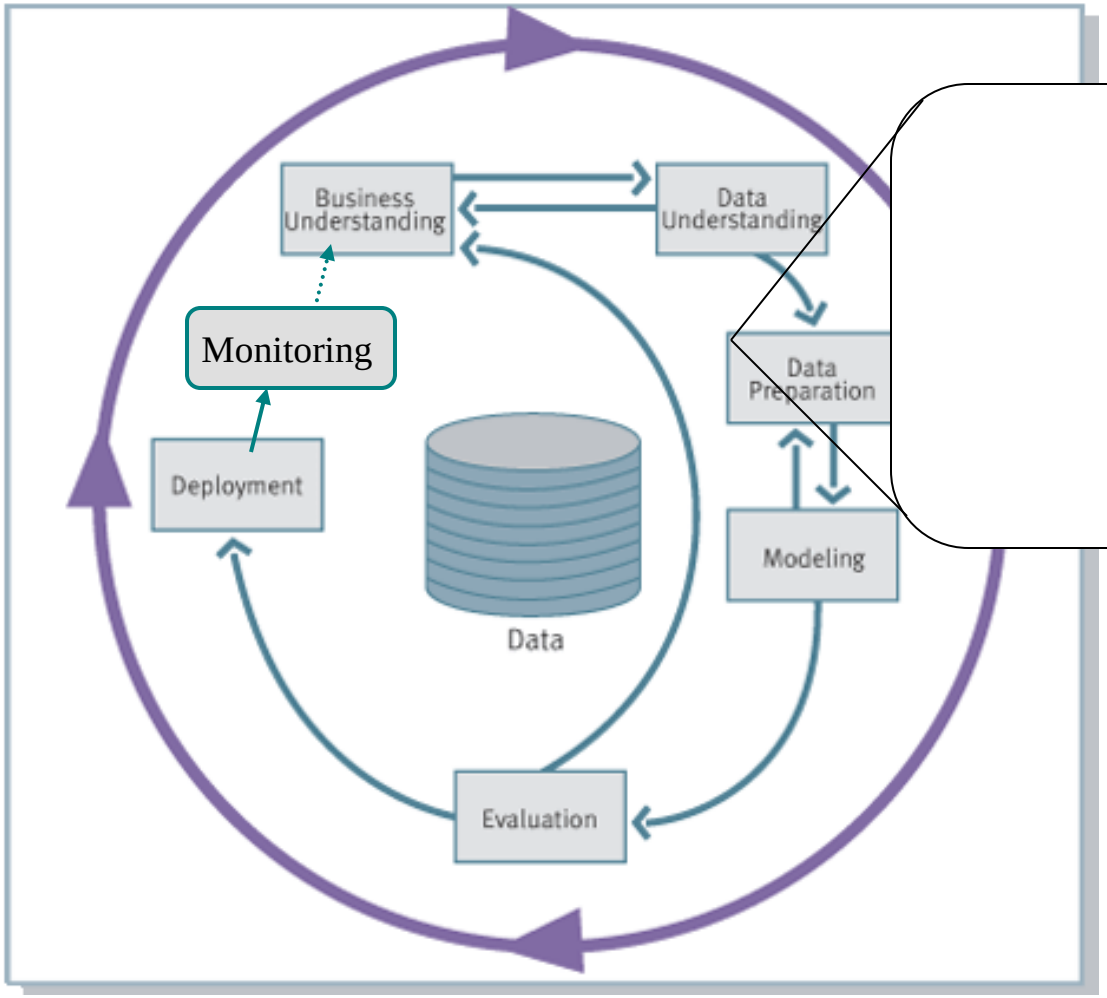
- Razumevanje podatkov
- Čiščenje podatkov
  - Metapodatki
  - Manjkajoče vrednosti
  - Podatki datumskega tipa
  - Nominalno → Numerično …
  - … in obratno = diskretizacija
- Izbor značilk in “napačni napovedovalci”
- Neuravnotežena porazdelitev razreda

Proces odkrivanja zakonitosti v podatkih – po CRISP-DM-u



glej
https://www.datascience-pm.com/crisp-dm-2/
za več informacij

Proces odkrivanja zakonitosti v podatkih – v praksi



Priprava podatkov

Priprava podatkov vzame (po nekaterih ocenah) 80 – 90% časa in napora

Razumevanje podatkov: relevantnost

- Kateri/kakšni podatki so sploh na voljo?
- So ti podatki sploh relevantni?
- So na voljo kakšni dodatni relevantni podatki?
- Koliko (zgodovinskih) podatkov je sploh na voljo?
- Kdo je strokovnjak za te podatke?

Razumevanje podatkov: kvantiteta

- Število primerov (zapisov)
  - *Ocena "čez palec": 5.000 ali več zaželeno*
  - Če manj: rezultati manj zanesljivi; uporaba posebnih prijemov (bagging, boosting …)
- Število atributov (značilk)
  - *Ocena "čez palec": za vsako značilko vsaj 10 primerov*
  - Če je značilk več: uporaba metod izbora/vzorčenja značilk
- Število vrednosti razrednega atributa
  - *Ocena "čez palec": > 100 primerov za vsako vrednost razreda*
  - Če neuravnoteženo: uporaba stratificiranega vzorčenja

Koraki čiščenja podatkov

- Pridobivanje podatkov in metapodatki
- Manjkajoče vrednosti
- Datumski format podatkov
- Pretvorba iz nominalnega v numerično
- Diskretizacija numeričnih podatkov
- Validacija podatkov in statistike

Čiščenje podatkov: pridobivanje

- Podatki so večinoma shranjeni v DBMS
  - ODBC, JDBC protokoli
- Podatki v "preprostih" tekstovnih datotekah
  - Fiksni (fixed-column) format
  - Razmejeni format: tabulator, vejica “,” , ostalo
    - npr. C4.5 in Wekin “arff” uporabljata CSV format
  - Pozor: pretvorba razmejevalnih znakov v nizih
- **Preveriti število atributov pred in po pretvorbi !!!**

Čiščenje podatkov: primer

- **Izvorni podatki (fiksni format)**

000000000130.06.19971979-10-3080145722   #000310 111000301.01.000100000000004
0000000000000.000000000000000.000000000000000.000000000000000.000000000000000.000000000000000.0000
00000000000. 000000000000000.000000000000000.0000000……
000000000000000.000000000000000.000000000000000.000000000000000.000000000000000.000000000000000.00
0000000000000.000000000000000.000000000000000.000000000000000.000000000000000.000000000000000.0000
00000000000.000000000000000.000000000000000.000000000000000.000000000000000.000000000000000.000000
000000000.000000000000000.000000000000000.000000000000000.00 000000000300.00 000000000300.00

- **Prečiščeni podatki (podatki razmejeni z vejico)**

0000000001,199706,1979.833,8014,5722   ,   ,#000310 ….
,111,03,000101,0,04,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0300,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0300,0300.00

Čiščenje podatkov: metapodatki

- **Tipi atributov:**
  - **binary, nominal (categorical), ordinal, numeric, …**
  - **For nominal fields: tables translating codes to full descriptions**
- **Vloga atributov:**
  - vhodni: vhod v algoritme modeliranja
  - ciljni: izhod algoritmov modeliranja
  - id/pomožni: obdržimo, a ne uporabljamo pri modeliranju
  - zanemarljivi: ne uporabljamo pri modeliranju
  - uteži: utežimo posamezne primere
  - …
- **Deskriptorji atributov (dodaten opis)**

Čiščenje podatkov: spremembe formata

Pretvorba v nek standardni format (npr. arff ali CSV)

- Manjkajoče vrednosti
- Datumski formati
- Diskretizacija numeričnih podatkov
- Odprava napak in osamelci
- Pretvorba urejenih nominalnih atributov v numerične

- ***V: Zakaj že?***

Čiščenje podatkov: spremembe formata (2)

Pretvorimo urejene nominalne atribute v numerične, da bi lahko kasneje uporabljali primerjalna operatorja “>” and “<” na vrednostih teh atributov.

Čiščenje podatkov: manjkajoče vrednosti

- Manjkajoče vrednosti – različni formati:
  - <prazno polje> “0” “.” “999” “N/A” …
- Standardizacija zapisa manjkajočih vrednosti (v WEKI = "?")

- ***V: Kako pa se spopademo z manjkajočimi vrednostmi?***

Čiščenje podatkov: manjkajoče vrednosti (2)

- Obravnava manjkajočih vrednosti:
  - izločimo primere, ki jih vsebujejo
  - izločimo atribute, ki jih vsebujejo > N%
  - obravnavamo kot ločene vrednosti (!)
  - nadomestimo s povprečjem, modusom …
  - prepustimo algoritmu strojnega učenja

Čiščenje podatkov: datumski formati

- Želja:
  interno predstaviti vse datume na enoten način
- Datumi lahko nastopajo v različnih formatih
  - npr. “Sep 24, 2003” , 9/24/03, 24.09.03 …
  - interno se datumi pretvorijo v neko standardno vrednost
- Najpogosteje zadostuje le leto (YYYY),
- Lahko pa potrebujemo še mesec, dan, uro …
- Predstavitev datumov kot YYYYMM ali YYYYMMDD je lahko OK, a imamo lahko težave
- ***V: Kakšne so lahko težave s predstavitvijo datumov v obliki YYYYMMDD?***

Čiščenje podatkov: datumski formati (2)

- Težava z datumi v YYYYMMDD obliki:
  - YYYYMMDD ne ohranja intervalov:
  - 20040201 – 20040131 (= 70)

    !=

    20040131 – 20040130 (= 1)
  - Kar lahko vnaša pristranskost v modele

Unificirani datumski formati

- Da bi ohranili intervale, lahko uporabimo:
  - Unix-ov sistem predstavitve datumov: število sekund od 1.1.1970 (00:00:00 UTC)
  - SAS-ov sistem: število dni od 1.1.1960
- Težava:
  - vrednosti niso intuitivne
  - ➔ ne pripomore k odkrivanju zakonitosti
  - ➔ oteženo preverjanje; večja verjetnost napak

KSP datumski format

$$\text{KSP datum} = \text{YYYY} + \frac{\text{"št\_dni\_od\_1.\_jan"} - 0{,}5}{365 + \text{"1\_če\_prestopno\_leto"}}$$

- Ohranja intervale (skoraj)
- Očitno v kateri četrtini leta je nek datum
  - Sep 24, 2003 = 2003 + (267–0,5)/365 = 2003,7301 (zaokrožimo na 4 decimalke)
- Konsistentno z dnevi, ki se začnejo ob poldne
- Lahko razširimo, da vsebuje tudi ure, minute …

Y2K težave: 2-cifrno leto

- 2-cifrna leta v starih podatkih – zapuščina Y2K
- npr. **V: Leto 02 – je to 1902 ali 2002?**
  - **O**: Odvisno od konteksta
    (npr. rojstni dan otroka ali letnica izgradnje hiše)
  - Tipičen pristop: MEJNO leto, npr. 30
  - če YY < MEJNO, potem 20YY, sicer 19YY

Pretvorbe: nominalno → numerično

- Nekateri ML algoritmi interno podpirajo nominalne atribute
- Nekateri drugi (nevronske mreže, regresija, najbližji sosedi) delujejo le z numeričnimi atributi
- Za uporabo slednjih moramo tako nominalne atribute pretvoriti v numerične
  - **V: Zakaj ne bi kar ignorirali nominalne atribute?**
  - O: Ker lahko le-ti vsebujejo koristne informacije
- Različni pristopi za binarne, urejene, več-vrednostne neurejene nominalne atribute

Pretvorbi: v razmislek

- Kako bi pretvorili binarni atribut v numeričnega?
  - npr. Spol = M, Ž
- Kako bi pretvorili urejene nominalne atribute v numerične?
  - npr. Ocene (po ZDA ocenjevalni lestvici)

Pretvorbe: binarno → numerično

- Binarni atributi
  - npr. Spol = M, Ž
- Pretvorimo v Atribut_0_1 z vrednostma 0 in 1
  - npr. Spol = M → Spol_0_1 = 0
    Spol = Ž → Spol_0_1 = 1

Pretvorbe: urejeno → numerično

- Urejene atribute (npr. Ocena) lahko pretvorimo v numerične in pri tem ohranimo *naravni* vrstni red vrednosti, npr.
  - A → 4,0
  - A- → 3,7
  - B+ → 3,3
  - B → 3,0
- **V: Zakaj je pomembno ohraniti *naravni* vrstni red vrednosti?**

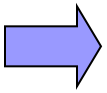
Pretvorbe: urejeno → numerično (2)

Naravni vrstni red omogoča smiselne primerjave, npr. Ocena > 3,5

Pretvorbe: nominalno, malo vrednosti

- Več-vrednostni, neurejeni atributi z malo vrednostmi (*čez palec* < *20*)
  - npr. Barva = Rdeča, Oranžna, Rumena, …, Vijolična
  - za vsako vrednost *v* ustvarimo binarno "označevalno" spremenljivko C_*v*, ki je 1, če Barva = *v*, 0 sicer

ID	Barva	…
371	Rdeča	
433	Rumena	



ID	C_rdeča	C_oranžna	C_rumena	…
371	1	0	0	
433	0	0	1	

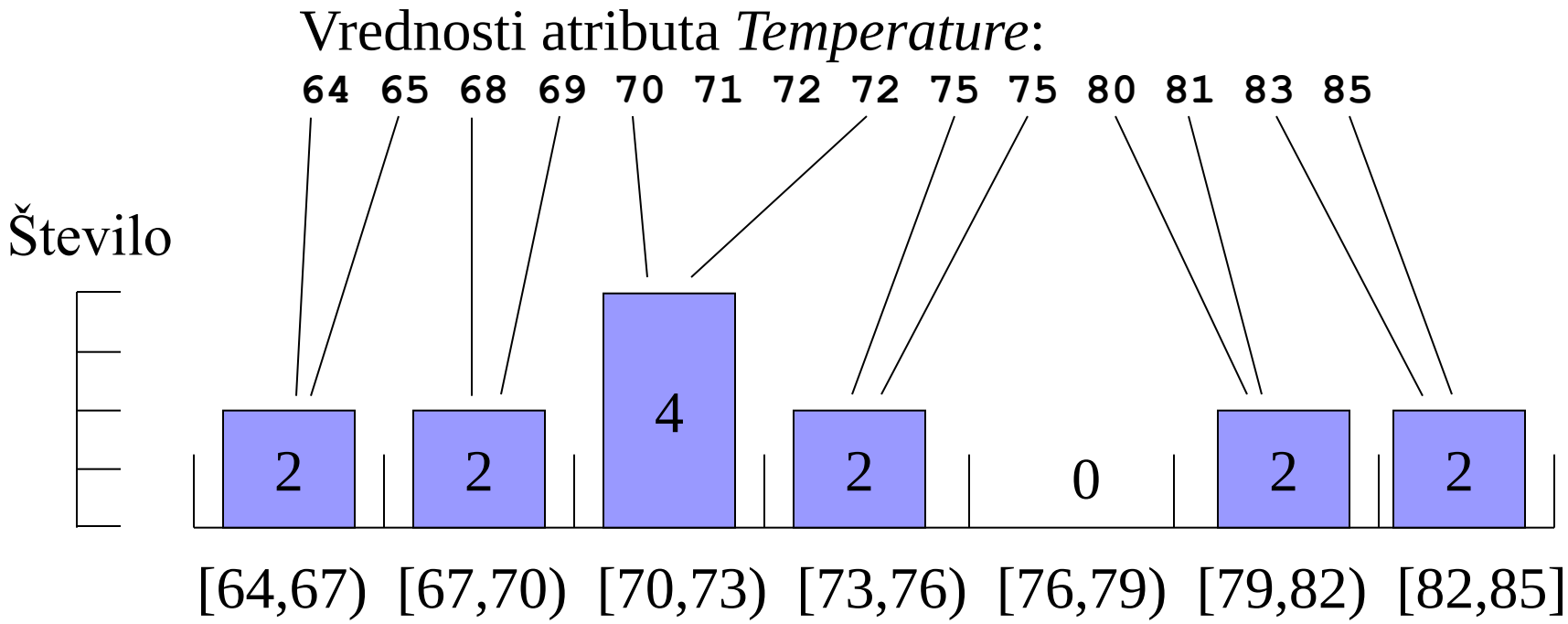
Pretvorbe: nominalno, veliko vrednosti

- Primeri:
  - Koda občine (212 vrednosti)
  - Koda poklica (7.000 vrednosti, le nekaj pogostih)
- **V: Kako se spopasti s takšnimi atributi?**
- O: Ignoriramo atribute tipa ID, katerih vrednosti so različne za vsakega od primerov
- Za ostale atribute, uporabimo “naravno” grupiranje:
  - npr. 212 kod občin → 12 statističnih regij
  - Poklici → izberemo najpogostejše, združimo preostale
- Postopamo kot pri pretvorbi nominalnih atributov z malo vrednostmi

Pretvorbe: diskretizacija

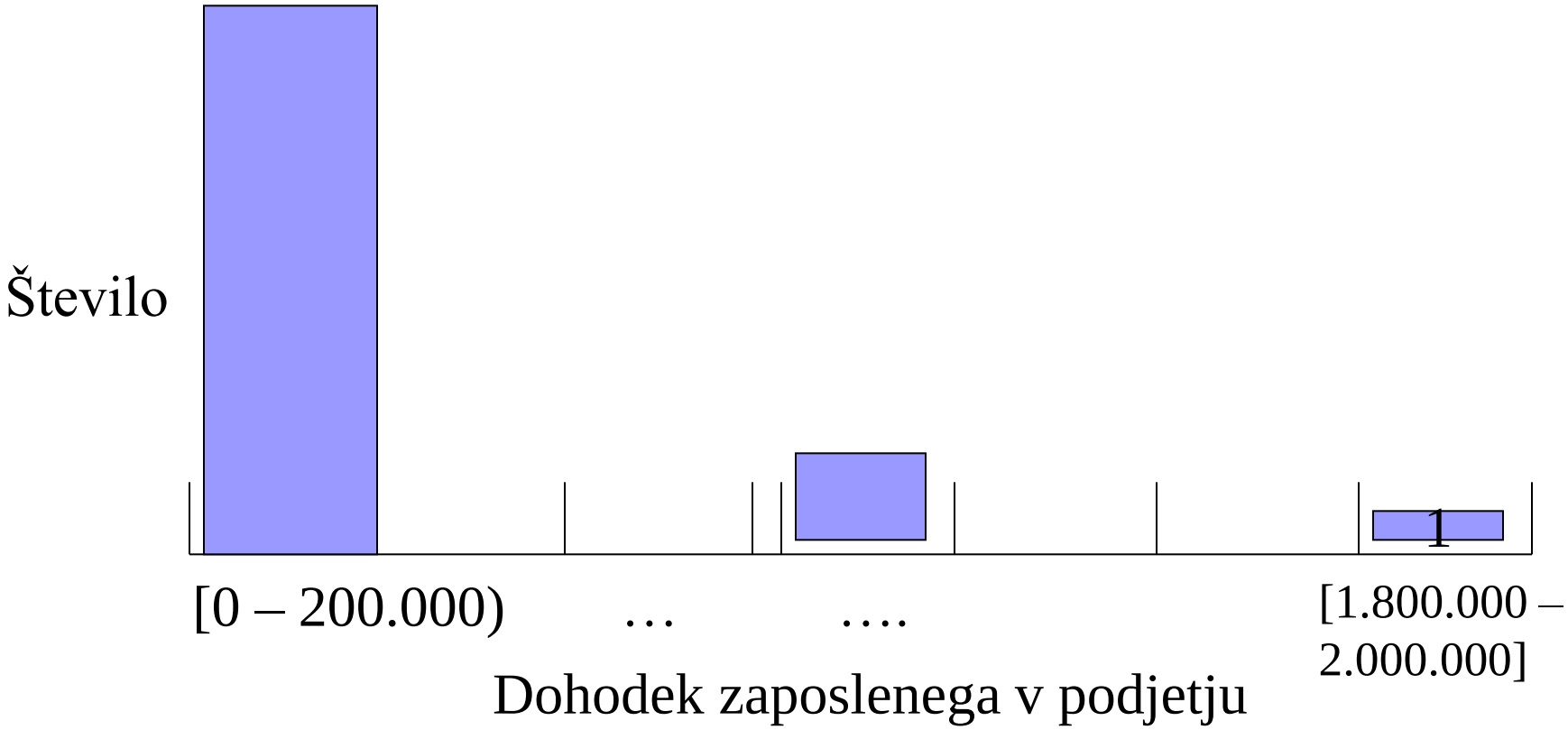
- Nekateri ML algoritmi zahtevajo diskretne vrednosti atributov,
  npr. večina različic Naivnega Bayesa …
- Diskterizacija je tudi uporabna za povzemanje podatkov
- Včasih ji tudi pravimo "predalčkanje" (ang. “binning”)

Diskretizacija: enake širine

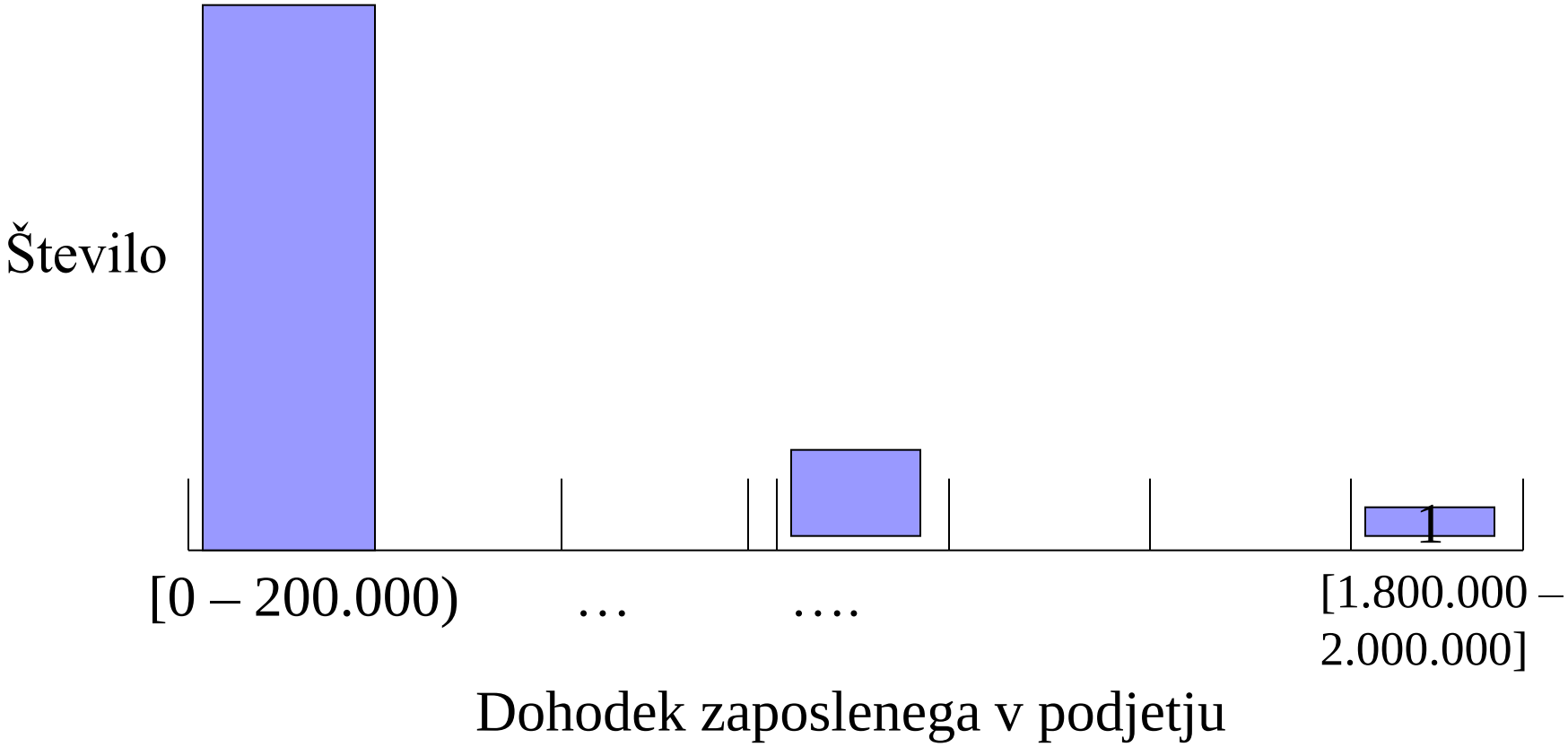


vsi "predalčki" so enako široki (3 enote)

Diskretizacija enakih širin lahko povzroči "luknje" v histogramu



Težava diskretizacije enakih širin

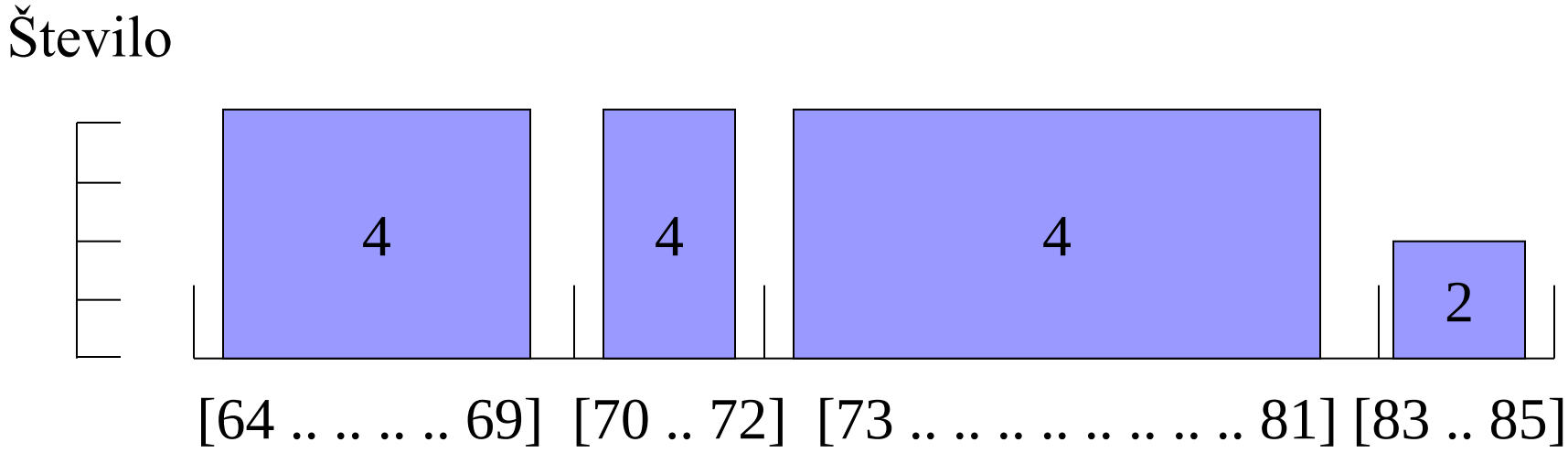


Kaj narediti, da bomo dobili enakomernejšo porazdelitev?

Diskretizacija: enake višine/frekvence

Vrednosti atributa *Temperature*:

64 65 68 69 70 71 72 72 75 75 80 81 83 85



enake višine (4), razen pri zadnjem "predalčku"

Diskretizacija: prednosti metode enakih višin

- V splošnem je boljša, ker ne generira "lukenj"
- V praksi: uporaba metode “skoraj enakih višin”, ki ne generira "lukenj" in uporablja bolj intuitivne razmejitve med vrednostmi ("predalčki")
- Dodatne opazke:
  - pogostih vrednosti ne delimo med "predalčke"
  - ločeni "predalčki" za posebne vrednosti (npr. 0)
  - "berljive" razmejitve med "predalčki" (zaokroževanje mejnih vrednosti)

Diskretizacija

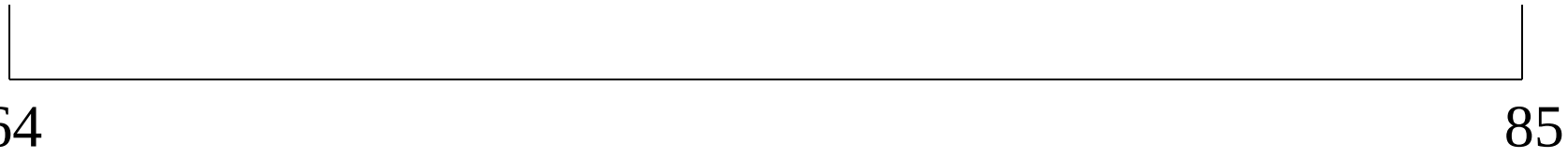
Obstaja morda še kakšen način?

Katero metodo še zasledimo v literaturi?

Diskretizacija: razredno-odvisna

min. št. vrednosti v "predalčku" = 3

64	**65**	**68**	**69**	**70**	**71**	**72**	**72**	**75**	**75**	**80**	**81**	**83**	**85**
Yes	No	Yes	Yes	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	No



Diskretizacija: zaključki

- D. enakih širin je najpreprostejša, uporabna v posebnih primerih
  - se zelo slabo obnese pri neenakomernih porazdelitvah
- D. enakih frekvenc da boljše rezultate
- Razredno-odvisna d. lahko da zelo dobre rezultate pri klasifikacijskih problemih
  - Opomba: odločitvena drevesa izvajajo d. med gradnjo
  - Naivni Bayes potrebuje že diskterizirane atribute
- Obstaja še vrsto drugih metod diskretizacije

Osamelci in napake

- Osamelci so vrednosti izven smiselnih okvirov (t.i. "out of range" vrednosti).
- Pristopi:
  - ne storimo ničesar
  - določimo zgornje in spodnje meje
  - uporabimo diskretizacijo

Čiščenje podatkov: uporaba statistik

```
*****************      Field 9:  MILES_ACCUMULATED

Total entries = 865636 (23809 different values).  Contains non-numeric values.    Missing
data indicated by "" (and possibly others).

Numeric items = 165161,  high = 418187.000,  low = -95050.000
       mean = 4194.557,  std = 10505.109,  skew =    7.000

Most frequent entries:

       Value   Total
             :         700474 ( 80.9%)
           0:         32748 (  3.8%)
           1:         416 (  0.0%)
           2:         337 (  0.0%)
          10:         321 (  0.0%)
           8:         284 (  0.0%)
           5:         269 (  0.0%)
           6:         267 (  0.0%)
          12:         262 (  0.0%)
           7:         246 (  0.0%)
           4:         237 (  0.0%)
```

Čiščenje podatkov: izbor atributov

Najprej: odstranimo atribute brez ali z minimalno variabilnostjo vrednosti

- Pregledamo št. različnih vrednosti atributa in le tega odstranimo
  - *Pravilo čez palec: če so skoraj vse vrednosti tega atributa enake (npr. null), razen morda minp % ali manj,*
  - *minp* je lahko 0,5% ali splošneje 5% od števila primerov najmanj pogostega razreda.

Napačni napovedovalci

- Napačni napovedovalci (ang. *False predictors*) so atributi, ki so močno korelirani z razredom, a opisujejo dogodke, ki so se zgodili *istočasno* ali *kasneje* kot dogodek opisan z razredom.
- Če PB ne beleži časa dogodkov, lahko napačni napovedovalec izpade kot dober napovedovalec.
- Primer: Datum odpovedi pogodbe je napačni napovedovalec prebega uporabnikov mob. storitev.

- **V: Morda še kak primer napačnega napovedovalca?**

Napačni napovedovalci: primer

V: Kaj je napačni napovedovalec, če napovedujemo verjetnost, da bo nek študent opravil izpit pri določenem predmetu?

O: Študentova končna ocena za ta predmet.

Napačni napovedovalci: iskanje “sumljivih” atributov

- Zgradimo odločitveno drevo
- Obravnavamo atribute z veliko napovedno močjo kot “sumljive”
  - velika napovedna moč = če atribut sam po sebi pojasni skoraj 100% klasifikacijske točnosti (atributi blizu korena drevesa)
- Preverimo “sumljive” atribute

  (dobro poznavanje problema ali domenski strokovnjak)
- Odstranimo napačni napovedovalec in postopek ponovimo

(skoraj) Avtomatsko odkrivanje lažnih napovedovalcev

- Za vsak atribut:
  - zgradimo 1-nivojsko odločitveno drevo
  - (ali izračunamo korelacijo z razredom)
- Rangiramo atribute po klasifikacijski točnosti (ali korelaciji z razredom)
- Atribute s točnostjo blizu 100% (opomba: meja je odvisna od domene) označimo kot "sumljive"
- Vse “sumljive” atribute preverimo z domenskim strokovnjakom

Izbor najrelevantnejših atributov

- Če je atributov "preveč", izberemo podmnožico najrelevantnejših.
- Lahko izberemo najboljših N glede na klasifikacijsko točnost (ali korelacijo).
- Kako izbrati primeren N?
  - Pravilo "čez palec" – obdržimo najrelevantvejših 50 atributov

Manjšanje št. atributov izboljša klasifikacijo

- Večina ML algoritmov išče nelinearne kombinacije atributov – kaj lahko se pojavijo nepravilne kombinacije, če je # primerov majhno, # atributov pa veliko
- Klasifikacijska točnost se (tipično) izboljša, če najprej zmanjšamo št. atributov
- Več-razredna hevristika: izberemo enako # atributov za vsak razred

Izpeljani atributi

- Bolje je imeti spodobno ML metodo in "dobre" atribute, kot pa imeti najboljšo ML metodo in "slabe" atribute.
- Zavarovalniški primer: moški so upravičeni do upokojitve in pokojnine pri 65 letih.
  ➔ Naredimo poseben binaren atribut!
- *Obstajajo naprednejše metode za avtomatsko preiskovanje kombinacij atributov, ampak so računsko zahtevne!

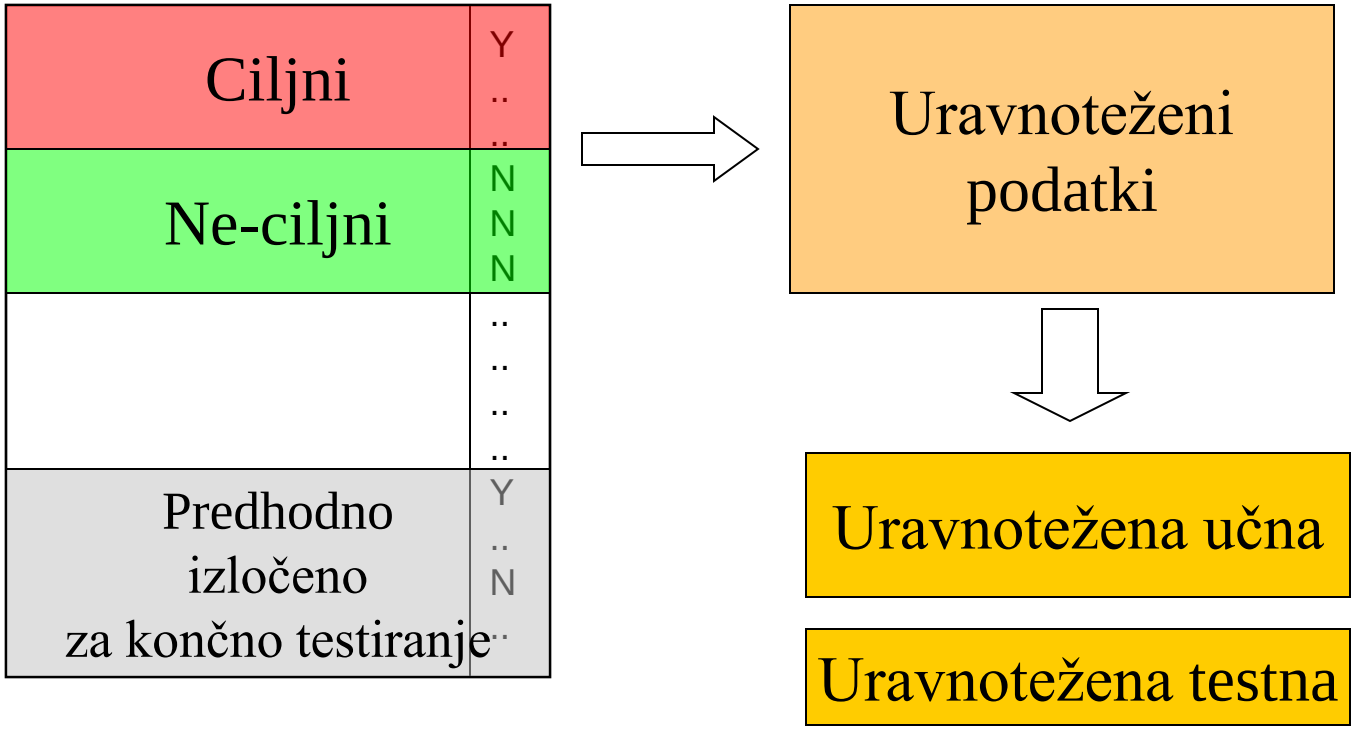
Neuravnotežena porazdelitev razreda

- Včasih so frekvence vrednosti razreda neenakomerne
  - Napoved prebegov: 97% ostane, 3% gre (na mesec)
  - Diagnoze v zdravstvu: 90% zdravih, 10% ima neko bolezen
  - eTrgovine: 99% ne kupi, 1% kupi
  - Varnost: > 99,99% Američanov ni teroristov
- Podobno tudi pri več-vrednostnih razredih
- Večinski klasifikator je lahko 97% točen (prebegi), a popolnoma neuporaben

Ravnanje z neuravnoteženimi podatki

- Za binarne razrede: naj bodo "pozitivne" vrednosti v manjšini
- Predhodno izločimo testne primere (npr. 30% podatkov)
  - izločene primere damo "na stran" in jih ne uporabimo do konca
- Med preostalimi primeri (npr. 70% podatkov) izberemo vse "pozitivne" primere
- Dodamo jim še (naključno izbranih) enako število "negativnih" primerov
- "Pozitivne" in "negativne" primere skupaj naključno premešamo in razdelimo na uravnoteženi učno in testno množico

Gradnja uravnoteženih množic



Učenje iz neuravnoteženih podatkov

- Gradimo modele na uravnoteženih učnih/testnih množicah
- Izvajamo končne ocene modelov na predhodno izločenih testnih podatkih

- "uravnoteževanje" lahko posplošimo na več razredov
  - stratificirano vzorčenje
  - zagotoviti moramo, da je vsak razred predstavljen v približno enakih razmerjih v učni in testni množici

Nekatere ključne ideje priprave podatkov

- Uporabimo metapodatke (če so na voljo)
- Preiščimo podatke za anomalije in napake
- Izločimo “napačne napovedovalce”
- Lahko še:
  - zmanjšamo število atributov
  - “uravnotežimo” podatke
- Preverimo vmesne rezultate po vsakem koraku priprave podatkov

Povzetek

Dobra priprava podatkov je ključnega pomena za gradnjo veljavnih in zanesljivih modelov