Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
Tržaška cesta 25, Ljubljana
Slovenija

Zoran Bosnić

Odskočna deska v okolje R

dodatno študijsko gradivo pri predmetu

Metode umetne inteligence

Ljubljana, marec 2010

Kazalo

BOSNIĆ, Zoran

Odskočna deska v okolje R: dodatno študijsko gradivo pri predmetu Metode umetne ingeligence

Ljubljana : Fakulteta za računalništvo in informatiko, 2010.

---

1. Uvod v okolje R

- http://www.r-project.org/
- nastavitev delovnega okolja: workspace, delovna mapa
- namestitev potrebnih paketov: tree, rpart, nnet, e1071, RWeka, randomForest
- odpiranje vhodne datoteke,
- nalaganje datoteke s programsko kodo, komentiranje z znakom #

1.1. Ukazi

`?funkcija`	odpre stran s pomočjo za funkcijo *funkcija*
`??funkcija` ali `help.search(funkcija)`	preišče vsebino vseh datotek s pomočjo za niz *funkcija*
`example(funkcija)`	pokaže primer uporabe funkcije *funkcija*
`ls()`	vrne seznam vseh objektov v delovnem okolju
`source("datoteka.R")`	naloži programsko kodo iz vhodne datoteke *datoteka.R*
`history(n)`	prikaže zgodovino ukazov (zadnjih `n` vrstic)

1.2. Osnovna aritmetika

`x <- 3`	priredi spremenljivki *x* vrednost 3
`x <- c(1,2,3,4)`	priredi spremenljivki *x* vektor s 4 elementi 1, 2, 3, 4
`y <- c(x, 0, x)`	priredi *y* vektor, ki je združen vektor dveh vektorjev *x*, ločenih z ničlo
`1/x`	izračuna *1/x* in prikaže rezultat na konzoli
`z <- 2*x + y + 1`	izračuna 2*x+y+1 in rezultat shrani v *z*
`1:12 ali seq(1,12)`	generira zaporedje (vektor) z elementi 1,2,3,4,5,6,7,8,9,10,11,12
`rep(3,5)`	generira vektor, ki vsebuje 5 ponovitev števila 3, torej: 3,3,3,3,3
`izbor <- x > 13`	izdela logični vektor (TRUE, FALSE), istoležen s komponentami *x*, ki so večje od 13, in ga shrani v *izbor*

1.3. Osnovni in kompleksni tipi spremenljivk

- **character**, **numeric**, **factor** – osnovni tipi (znakovni, številčni in diskretni – ta ima vnaprej določen nabor vrednosti), npr.:

```
#primer spremenljivke tipa factor:
> dnevi <- c("petek", "torek", "petek", "sreda", "torek")
> dnevi.factor <- as.factor(dnevi)
> dnevi.factor
[1] petek torek petek sreda torek
Levels: petek sreda torek
```

- **vector** – enorazsežni vektor (vse vrednosti so istega tipa), npr.

```
> a <- c("a","b","c",1)    #R avtomatsko pretvori zadni element
> a
[1] "a" "b" "c" "1"
```

- **matrix** – večdimenzionalni vektorji, vsi elementi so enakega osnovnega tipa (character, numeric ali factor),

- **data.frame** – podatkovna zbirka vsak stolpec lahko vsebuje spremenljivke svojega tipa (primerno za različne tipe atributov!)

1.4. Funkcije za delo s podatkovnimi tipi

`as.vector(argument)`
`as.matrix(argument)`
`as.data.frame(argument)`
`as.numeric(argument)`
`as.factor(argument)`
`as.character(argument)`
	pretvori `argument` v tip, ki ga določa pretvorbeni klic

`m <- matrix(nrow=10, ncol=3)` `dim(m), nrow(m), ncol(m)`	izdelamo matriko z 10 vrsticami in 3 stolpci vrne dimenzijo matrike, število vrstic, število stolpcev
`rbind(m1, m2)`	združi `m1` in `m2` po vrsticah (imeti morata enako število stolpcev)
`cbind(m1, m2)`	združi m1 in `m2` po stolpcih (imeti morata enako število vrstic)
`table(vec1, vec2)`	navzkrižno tabelira (prikaže frekvence) vrednosti dveh vektorjev
`data.frame(col1=..., col2=...)`	izdela podatkovno zbirko s stolpci `col1`, `col2`, ...
`head(data.frame)`	vrne začetne vrstice podatkovne zbirke
`names(data.frame)`	vrne imena stolpcev podatkovne zbirke

1.5. Naslavljanje elementov znotraj podatkovnih struktur

`x[c(1,5,2)]`	naslavljanje z indeksi (izberemo 1., 5. in 2. element)
`x[x>10]`	naslavljanje z logičnim pogojem (izberemo elemente, ki so večji od 10)
`x[-c(1,5,2)]`	elemente izločamo z negativnimi indeksi (izberemo vse elementa razen 1., 5. in 2.)
`x[c('tretji','prvi')]`	naslavljanje vektorja z imeni komponent (poimenujemo jih z `names(x) <-` ...)
`t[c(1,5)] <- c(2,3)`	naslovljenim delom vektorja lahko priredimo vrednosti
`m[vrstica, stolpec]`	naslavljanje matrike s številko vrstice in stolpca
`m[,stolpec]`	naslavljanje (vseh vrstic in) celega stolpca
`m[vrstica,]`	naslavljanje določene vrstice (in celega stolpca)
`data.frame[,'spol']` `data.frame$spol`	naslavljanje stolpcev podatkovne zbirke po imenu (npr. `spol`)
`which(vektor)`	vrne indekse, za katere je vrednost logičnega vektorja `vektor` enaka `TRUE`

1.6. Pogoste funkcije

`log, exp, sin, cos, tan, sqrt,` `sum, mean, var, min, max`	matematične funkcije
`length(x)`	dolžina vektorja `x`
`sort, order`	`sort` sortira elemente; `order` vrne vrstni red (rang) elementov
`sample(x, n, replace=T)`	generira `n` števil (s ponavljanjem, če `replace=T`) z intervala [1,`x`]
`rnorm(n, mean = 0, sd = 1)`	generira `n` števil, ki so porazdeljena po Gaussovi porazdelitvi graf gostote verjetnosti: `plot(density(rnorm(100)))`
`runif(n, min=0, max=1)`	generira `n` števil, ki so porazdeljena po enakomerni porazdelitvi `plot(density(runif(10000)))`

Primer 1: Generiranje umetnega problema (diskretni atributi, problem XOR)

```
# regresijski problem   (XOR)
#določimo vrednosti atributov in ciljne vrednosti (R)
a1 <- as.factor(sample(c(0,1), 1001, replace = T))
a2 <- as.factor(sample(c(0,1), 1001, replace = T))
a3 <- as.factor(sample(c(0,1), 1001, replace = T))
R <- ifelse(a1 != a2, 1, 0)
#združimo vse v tempData
tempData <- data.frame(a1 = a1, a2 = a2, a3 = a3, R = R)
# razdelimo na učno (ima 1000 primerov) in testno množico (samo 1 primer)
ucna <- tempData[1:1000,]
primer <- tempData[1001,]
```

2. Nadzorovano učenje v okolju R

- vgrajene podatkovne zbirke: `help(package="datasets")`
- UCI ML repozitorij podatkov: http://archive.ics.uci.edu/ml/ (ali išči z Google-om niz "UCI ML"),

2.1. Branje tekstovnih datotek

Za branje zunanjih datotek uporabimo funkcijo `read.table`, ki ima naslednjo (osnovno) sintakso (za več glej `?read.table`):

```
read.table(file,
      header = FALSE,          #ali imamo vrstico z imeni strolpcev
      sep = "",                #presledek med vrednostmi ("\t" je tabulator)
      quote = "\"'",           #simbol za navednice
      dec = ".",               #simbol za decimalno piko
      row.names,               #imena vrstic
      col.names,               #imena stolpcev
      colClasses = NA,         #tipi vhodnih spremenljivk
      na.strings = "NA",       #manjkajoče spremenljivke
      nrows = -1,              #največ dovoljeno vrstic za branje (-1 = max)
      skip = 0,                #koliko vrstic na začetku naj preskoči
      comment.char = "#",      #vrstice, ki jih ignoriramo
)
```

Primer: branje datoteke auto-mpg.data iz UCI repozitorija.

Opisi atributov (auto-mpg.names) so naslednji:

1. mpg: continuous
2. cylinders: multi-valued discrete
3. displacement: continuous
4. horsepower: continuous
5. weight: continuous
6. acceleration: continuous
7. model year: multi-valued discrete
8. origin: multi-valued discrete
9. car name: string (unique for each instance)

```
podatki <- read.table("auto-mpg.data.txt",
      colClasses = c('numeric','numeric',rep('numeric',4),"numeric", "numeric", "character"),
      col.names=c('mpg','cylinders','dis','horse','weight','accel','year','origin','name'),
      na.strings="?")
```

2.2. Pogoste funkcije za opisovanje in obdelavo podatkovnih zbirk

`str(podatki)`	vrne kratek opis STRukture podatkovne zbirke
`summary(podatki)`	izračuna kratke statistike atributov
`complete.cases(podatki)`	vrne logični vektor s komponentami, ki predstavljajo, ali ima primer vse vrednosti znane
`podatki[complete.cases(podatki),]`	izbere samo vrstice, ki imajo vse znane vrednosti atributov

2.3. Klasifikacijski in regresijski modeli, napovedovanje

Modeliranje in napovedovanje:

- napovedni model zgradimo s sintakso:
  `funkcija(formula, ucna.mnozica)`
- napovedi za testne primere izračunamo s sintakso:
  `predict(model, testna.mnozica)`
- `formula` je zapis odvisnosti med spremenljivkami, katero želimo, da model modelira (npr., če želimo iz vseh preostalih atributov napovedovati vrednost atributa z imenom `mpg`, zapišemo kot formulo `mpg ~ .` (beri: modeliraj `mpg` v odvisnosti (znak `~`) od vseh (znak `.`) ostalih atributov)

Učno in testno množico je za klasifikacijo in regresijo potrebno pripraviti drugače, in sicer:

- razred, ki ga napovedujemo pri klasifikacijskem problemu, mora biti v podatkovni zbirki tipa **`factor`** (torej diskretna spremenljivka)
- označba, ki jo napovedujemo pri regresijskem problemu, mora biti v podatkovni zbirki tipa **`numeric`** (torej navadna številska spremenljivka)

Klasifikacijski in regresijski modeli:

- primeri funkcij klasifikacijskih modelov: `rpart` (klasifikacijsko drevo), `svm`, `nnet`, `naiveBayes`, `IBk` (k najbližjih sosedov – iz razreda `RWeka`),
- primeri funkcij regresijskih modelov: `rpart` (regresijsko drevo), `lr` (linearna regresija), `svm`, `nnet`, randomForest

Primer 1: gradnja klasfifikatorja kNN

```
#denimo, da imamo podano množico podatkov z imenom dataset, ki ima diskretni razred age
#razdelimo dataset na učno in testno množico
testna.idx <- sample(nrow(dataset), 15) #naključno izberemo 15 testnih primerov
testna <- dataset[testna.idx, ]
ucna <- dataset[-testna.idx, ]

#zgradimo klasifikator
library(RWeka) #potrebno za funkcijo IBk
model.svm <- svm(age ~ ., ucna)])  #age je faktorska spremenljivka
#modelu lahko opcijsko nastavimo tudi število sosedov: control = Weka_control(K = 5)

#napovemo vrednosti testnim primerom
napovedi <- svm(model.svm, testna)
```

Primer 2: gradnja regresorja randomForest

```
#zgradimo regresor
library(randomForest) #potrebno za funkcijo randomForest
model.rf <- randomForest(age.cont ~ ., ucna)  #age.cont je numerična spremenljivka
#napovemo vrednosti testnim primerom
napovedi <- svm(model.svm, testna)
```

2.4. Generične funkcije za delo z modeli

Spodaj so naštete funkcije, ki so implementirane za večino napovednih modelov. Te funkcije se odzivajo drugače glede na vrsto modela, ki ga podamo kot argument. Ni nujno, da je za vsak model implementirana vsaka od spodaj naštetih funkcij.

`print(model)`	izpiše strukturo ali vsebino modela
`predict(model, data)`	napove vrednosti testnim primerom `data` z uporabo modela `model`
`plot(model)`	grafično prikaže model
`summary(model)`	poda jedrnat opis modela
`formula(model)`	vrne formulo, s katero je bil zgrajen model
`deviance(model)`	vrne razlike med pravimi vrednostmi razredov / označb primerov in napovedmi modela za iste primere

2.5. Ocenjevanje učenja

Klasifikacijska točnost

Je mera za ocenjevanje kakovosti klasifikacijskega modela. Definirana je kot delež primerov, ki so klasificirani pravilno. Denimo, da imamo:

- testno množico z imenom testna
- razred z imenom razred
- napovedi razreda z imenom napovedi

Klasifikacijsko točnost izračunamo z izrazom

```
length(which(testna$razred == napovedi)) / length(napovedi)
```

ali lepše, če izdelamo funkcijo `klas.tocnost`:

```
klas.tocnost <- function(razredi, napovedi) {   #formalni argumenti funkcije
        odstotek <- length(which(razredi == napovedi)) / length(napovedi)
        return(odstotek)
}
```

in klasifikacijsko točnost ocenimo s klicem:

```
klas.tocnost(testna$razred, napovedi)
```

Relativna srednja kvadratna napaka

Je mera za ocenjevanje kakovosti regresijskega modela. Definirana je kot razmerje med (i) srednjo kvadratno napako razlik med napovedmi in pravimi regresijskimi označbami ter (i) srednjo kvadratno napako referenčnega modela, ki napoveduje povprečno vrednost označb na učni množici.

Relativno srednjo kvadratno napako (rmse) lahko izračunamo s funkcijo:

```
rmse.napaka <- function(oznacbe, napovedi, oznacbe.ucna)
{
        rmse <- sum((oznacbe-napovedi)^2)/ sum((oznacbe-mean(oznacbe.ucna))^2)
        return(rmse)
}
```

2.6. Programski konstrukti

if (pogoj) stavek1 else stavek2	pogojni stavek
while (pogoj) stavki	zank
for (var in seznam) stavki	zanka

imefun <- function(arg1, arg2, ...) {
koda
}
	deklaracija funkcije

Domača naloga: Napiši funkcije za ostale mere za ocenjevanje kakovosti učenja in jih uporabi v domači nalogi (senzitivnost, specifičnost, preciznost, priklic itd.)!

3. Vizualizacija in grafika v okolju R

3.1. Primeri vizualizacij s spleta

Primeri grafov na : http://www.stat.auckland.ac.nz/~paul/RGraphics/rgraphics.html

3.2. Funkcije za vizualizacijo podatkov

Primarne funkcije (ustvarijo novo grafično okno)

UKAZ	PRIMER	POMEN
`plot(x, y)`	`plot(x$horse, x$accel)`	nariše graf točk vektorja x proti y
`coplot(a ~ b | c +d )`	`coplot( x$horse ~ x$accel | x$year)`	serija grafov a=f(b), pogojevanih z vrednosmi spremenljivk c in d
`pairs(dataset)`	`pairs(quake)` (knjižnica rpart)	serija grafov parov vseh atributov
`hist(vector)`	`hist(t$year, prob=T)` `lines(density(t$year))`	histogram vektorja

Pomožne grafične funkcije (naknadno dorisujejo v obstoječe grafično okno)

`points(x,y)`	v graf doriše množico točk `(x,y)`
`lines(x,y)`	točke, podane z `(x,y)` poveže z daljicami
`text(x, y, besedilo)`	na koordinate `x` in `y` napiše besedilo
`abline(a, b)`	doda premico s presečiščem `a` in strmino `b`
`polygon(x,y)`	izriše poligon z oglišči v točkah `(x,y)`
`axis(side)`	označi osi na strani `side` grafa (vrednosti pomenijo: `1`-spodaj, `2`, levo, `3`-zgoraj, `4`-desno)
`rect(xleft, ybottom, xright, ytop, density = NULL, angle = 45)`	izriše pravokotnih na koordinatah s senčenjem, ki ga opredeljujeta `density` in `angle`

3.3. Argumenti funkcij in nastavitve splošnih grafičnih parametrov

Naslednje argumente lahko podamo pri večini grafičnih funkcij iz točke 4.2.

`type=`	način izrisovanja podatkov. Kot vrednost lahko podamo "p" (points), "l" (lines), "b" (both=points+lines).
`xlab=, ylab=`	oznaka vodoravne in navpične osi
`main=`	naziv grafa
`sub=`	podnaslov grafa
`pch=`	znak, ki se uporablja za izris točke na grafu (point character). Lahko direktno podamo znak (npr. "+") ali pa številsko vrednost 0-25 za izris posebnih znakov (glej dokumentacijo)
`lty=`	vrsta črte (line type)
`lwd=`	debelina črte (line width)
`col=`	barva
`cex=`	faktor povečave znakov (character expansion)

Nastavitve privzetih grafičnih parametrov lahko tudi pregledujemo in spremenimo njihove vrednosti z ukazom `par()`. Ta ukaz prikaže vrednosti vseh parametrov. Privzete vrednosti lahko (trajno) spremenimo z ukazom:

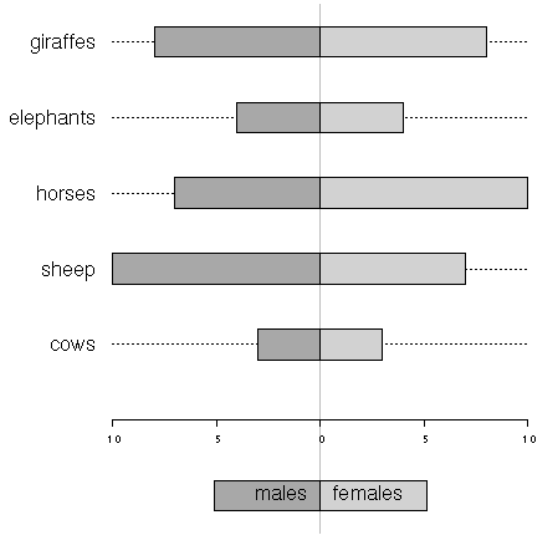
```
stari.par <- par(name=setting) # pri tem se v stari.par starijo predhodne vrednosti
```

Primer:

```
stari <- par(mfrow=c(3,2))      # izrisuje grafe v mreži oblike 3x2
```

Primer: primer vizualizacije (spletni vir)

```
groups <- c("cows", "sheep", "horses", "elephants", "giraffes")
males <- sample(1:10, 5)
females <- sample(1:10, 5)

par(mar=c(0.5, 5, 0.5, 1))

plot.new()
plot.window(xlim=c(-10, 10), ylim=c(-1.5, 5.5))

ticks <- seq(-10, 10, 5)
y <- 1:5
h <- 0.2

lines(rep(0, 2), c(-1.5, 5.5), col="grey")
segments(-10, y, 10, y, lty="dotted")
rect(-males, y-h, 0, y+h, col="dark grey")
rect(0, y-h, females, y+h, col="light grey")
mtext(groups, at=y, adj=1, side=2, las=2)
par(cex.axis=0.5, mex=0.5)
axis(1, at=ticks, labels=abs(ticks), pos=0)

tw <- 1.5*strwidth("females")
rect(-tw, -1-h, 0, -1+h, col="dark grey")
rect(0, -1-h, tw, -1+h, col="light grey")
text(0, -1, "males", pos=2)
text(0, -1, "females", pos=4)
box("inner", col="grey")
```



Moji zapiski...