STROJNO UČENJE

2. Domaća Zadaća

Krešimir Špes
0036419866
ak. god. 2011. / 2012.

Naivni Bayes:

Emprijska pogreska: 6.496%

Pogreska generalizacije: 8.15%

Naivni Bayes uz Laplaceovo zaglađivanje:

Emprijska pogreska: 6.689%

Pogreska generalizacije: 7.262%

Vrijednosti su izrčunate i uprosiječene na 1000 eksperimenata u kojima se svaki puta nasumično presloži skup podataka i razdijeli.

Kao što možemo vidjeti, uz laplaceovo zaglađivanje dobivamo nešto manje pogreške generalizacije što je i logčino jer u skupu podataka ima vjerojatnosti koje su 0

evo jedan primjer izračunatih vjerojatnosti u programu (1 eksperiment):

Naivni Bayes:

setosa(1)	setosa(2)	setosa(3)
0.794	0.206	0.000
0.000	0.235	0.765
1.000	0.000	0.000
1.000	0.000	0.000

versi(1)	versi(2)	versi(3)
0.094	0.594	0.312
0.531	0.312	0.156
0.000	0.938	0.062
0.000	0.906	0.094

virgi(1)	virgi(2)	virgi(3)
0.029	0.235	0.735
0.412	0.353	0.235
0.000	0.059	0.941
0.000	0.059	0.941

Emprijska pogreska: 6.0%
Pogreska generalizacije: 10.0%

Naivni Bayes uz Laplaceovo zaglađivanje:

setosa(1)	setosa(2)	setosa(3)
0.743	0.229	0.029
0.029	0.229	0.743
0.943	0.029	0.029
0.943	0.029	0.029

versi(1)	versi(2)	versi(3)
0.135	0.649	0.216
0.541	0.324	0.135
0.027	0.865	0.108
0.027	0.811	0.162

virgi(1)	virgi(2)	virgi(3)
0.027	0.270	0.703
0.324	0.378	0.297
0.027	0.054	0.919
0.027	0.054	0.919

Emprijska pogreska: 6.0%
Pogreska generalizacije: 8.0%

Vidi se da su neke vrijednosti u naivnom bayesu == 0, dok su u zaglađenom bayesu malo veće od nule. (napomena vjerojatnosti s lijeva i desna nisu na istom skupu jer se set podataka nasumično mješa)

U nastavku je dan ispis python koda za ovaj zadatak (također dan kao zasebna datoteka):

```
import random

data=[]
p1 = []
p2 = []
p3 = []

nEksperimenata = 1

def log(s):
	#pass
	print s

def ml(stupac, vrijednost, klasa):
	nKlasa = 0
	nVrijednost = 0

	for i in data:
	 if i[4] == klasa:
		nKlasa += 1
		if i[stupac] == vrijednost: nVrijednost += 1

# naivni bayes
	if nKlasa == 0: return 0
	return float(nVrijednost) / nKlasa
# bayes sa laplaceovim zagladjivanjem
	#return float(nVrijednost + 1) / (nKlasa + 3)



# parsaj podatke
f = open('dataset.txt','r')

for line in f:
	lst = line.split(',')
	data.append( [ float(lst[0]), float(lst[1]), float(lst[2]), float(lst[3]), lst[4].replace('\n','').replace('Iris-','') ] )
f.close()

def diskretiziraj(stupac):
	lst = []

	for i in data:
	 lst.append(i[stupac])

	lst.sort()

	granica1 = lst[int(  len(lst)/3)]
	granica2 = lst[int(2*len(lst)/3)]

	for i in data:
	 if    i[stupac] < granica1: i[stupac] = 0
	 elif  i[stupac] < granica2: i[stupac] = 1
	 else: i[stupac] = 2
```

```
for i in range(0,4):
	diskretiziraj(i)

def greska(set):
	nKrivih = 0
	for i in set:
	 a = p1[0][i[0]] * p1[1][i[1]] * p1[2][i[2]] * p1[3][i[3]]
	 b = p2[0][i[0]] * p2[1][i[1]] * p2[2][i[2]] * p2[3][i[3]]
	 c = p3[0][i[0]] * p3[1][i[1]] * p3[2][i[2]] * p3[3][i[3]]
	 if   a > b and a > c: y = 'setosa'
	 elif b > c:           y = 'versicolor'
	 else:                 y = 'virginica'
	 if y != i[4]:
		log(str(a) + "," + str(b) + "," + str(c) + "," + str(y) + " ---> " + repr(i))
		nKrivih += 1

	return float(nKrivih) / len(set)

empirijska = 0
generalizacijska = 0
for i in range(0,nEksperimenata): # napravi 1000 eksperiminata radi tocnijih mjerenja
	p1 = []
	p2 = []
	p3 = []
	random.shuffle(data)

	validation_set = data[2*len(data)/3:]
	data = data[:2*len(data)/3]


	log('---------------------------------------')
	log('setosa(1)\tsetosa(2)\tsetosa(3)')
	log('---------------------------------------')
	for i in range(0,4):
	 p1.append( [ ml(i, j, 'setosa') for j in range(0,3) ] )
	 log("%.3f\t\t%.3f\t\t%.3f" % (p1[i][0], p1[i][1], p1[i][2]))

	log('---------------------------------------')
	log('versi(1)\tversi(2)\tversi(3)')
	log('---------------------------------------')
	for i in range(0,4):
	 p2.append( [ ml(i, j, 'versicolor') for j in range(0,3) ] )
	 log("%.3f\t\t%.3f\t\t%.3f" % (p2[i][0], p2[i][1], p2[i][2]))

	log('---------------------------------------')
	log('virgi(1)\tvirgi(2)\tvirgi(3)')
	log('---------------------------------------')

	for i in range(0,4):
	 p3.append( [ ml(i, j, 'virginica') for j in range(0,3) ] )
	 log("%.3f\t\t%.3f\t\t%.3f" % (p3[i][0], p3[i][1], p3[i][2]))
	log('---------------------------------------')
	empirijska += greska(data)*100
	log('---------------------------------------')
```

```
        generalizacijska += greska(validation_set)*100
        log('----------------------------------------')
        data += validation_set

print 'Emprijska pogreska:      ' + str(empirijska/nEksperimenata) + '%'
print 'Pogreska generalizacije: ' + str(generalizacijska/nEksperimenata) + '%'
```