Druga domača naloga

Anže Pečar (63060257)

5. marec 2012

1 Uvod

Naša naloga je bila izračunati kako informativne so značilke za posamezen razred. Podatke smo najprej binarilizirali in nato izračunali medsebojno informacijo. Na koncu smo s pomočjo permutacijskega testa določili kateri atributi so značilni za posamezen razred. Delali smo na zmanjšanem naboru podatkov, ki jih uporablja tekmovanje *JRS 2012 Data Mining Competition: Topical Classification of Biomedical Research Papers*.

2 Metode

2.1 Količina medsebojne informacije

Na predavanjih smo količino medsebojne informacije definirali kot

$$I(X;Y) = H(Y) - H(Y|X)$$

pove pa nam koliko informacije prispeva posamezen atribut. H predstavlja entropijo, ki je definirana kot:

$$H(A) = p * log_2 \frac{1}{p}$$

Za primer bomo izračunali medsebojno informacijo za razred $c40$ in atribut D_0. Razred $c40$ ima 1498 vrednosti F in 502 vrednosti T, kar pomeni, da je njegova entropija enaka

$$H(c40) = H(\frac{1498}{2000}, \frac{502}{2000}) = 0.812859243$$

Za atribut D_0 pa je razporejen po naslednji tabeli:

1489	$D_0 = 0$ in $c40 = F$
501	$D_0 = 0$ in $c40 = T$
9	$D_0 = 1$ in $c40 = F$
1	$D_0 = 1$ in $c40 = T$

S pomočjo zgornje tabele lahko izračunamo $H(c40|D_0)$:

$$H(c40|D_0) = \frac{10}{2000} * H(\frac{9}{2000}, \frac{1}{2000}) + \frac{1990}{2000} * H(\frac{1489}{2000}, \frac{501}{2000}) = 0.812322554$$

Količino medsebojne enakosti dobimo tako, da dobljeni vrednosti odštejemo:

$$I(D_0; c40) = H(c40) - H(c40|D_0) = 0.000536689$$

Če nad istimi podatki poženemo še funkcijo InformationGain iz paketa Orange dobimo podoben rezultat:

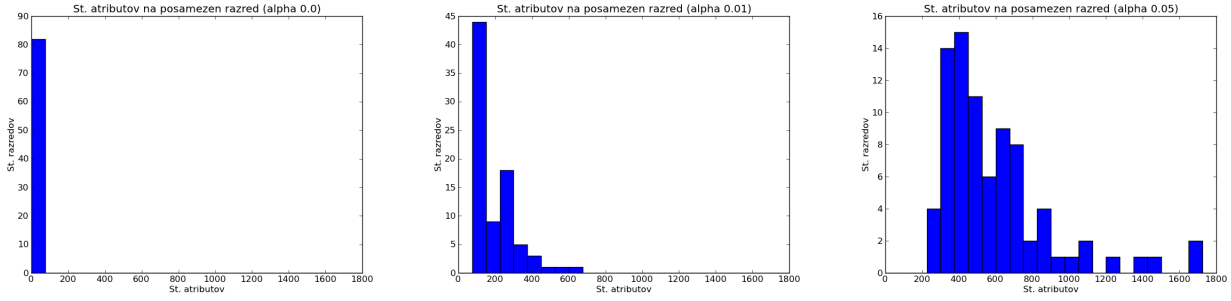
$$I(D_0; c40) = 0.0005311369895935059$$

Izračun količine medsebojne informacije nam sam po sebi še ne pove ali je atribut povezan z razredom. Zato uporabimo tako imenovani **permutacijski test**, ki poteka tako, da naredimo določeno število naključnih permutacij vrednosti razreda. Za te naključne permutacije izračunamo InformationGain. Če je odstotek tako dobljenih vrednosti večji od vrednosti *Alphe* našo hipotezo, da je atribut povezan z razredom, lahko zavrnemo. Če pa je odstotek tako dobljenih vrednosti manjši *Alphe* pa lahko zavrnemo obratno hipotezo, da atribut in razred nista povezana.

3 Rezultati

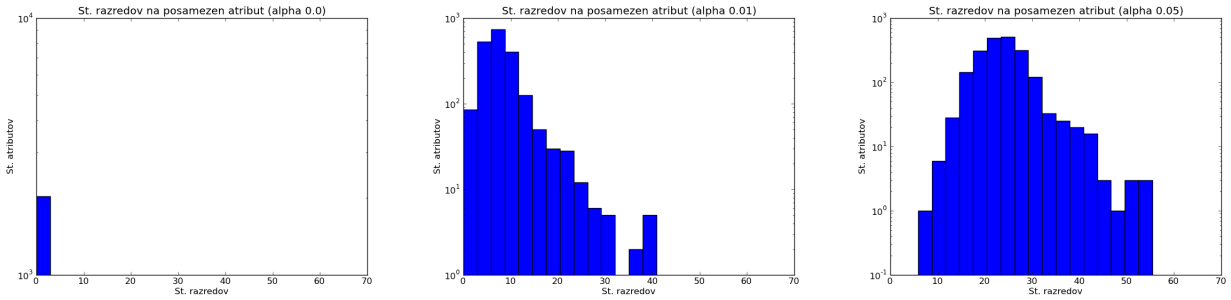
3.1 Grafi

Delovanje svojega algoritma sem najprej preizkusil nad alpha vrednostjo 0.0. Rezultati so vidni na Slikah 1 in 2 - prvi graf. Zanimalo me je tudi, kako vpliva velikost alphe na informativnost značilk. Algoritem sem zato pognal za alpho vrednost 0.01 in 0.05. Rezultati so na Slikah 1 in 2 drugi in tretji graf.



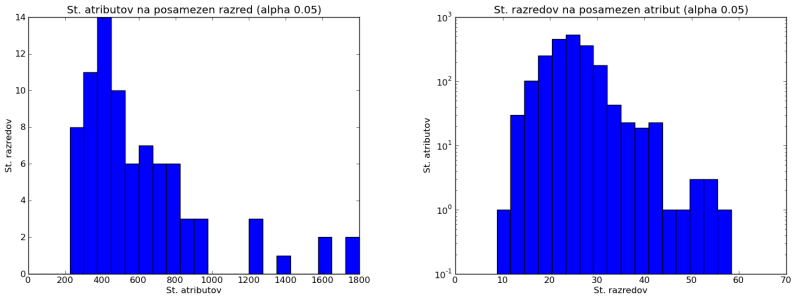
Slika 1: Rezultati 100 permutacij za različne vrednosti Alpha

Slika 1 prikazuje število atributov na posamezen razred za različne vrednosti alpha (0.0, 0.01 in 0.05).



Slika 2: Rezultati 100 permutacij za različne vrednosti Alpha, logaritemska skala

Slika 2 pa prikazuje št. razredov na posamezen atribut. Za lepši prikaz sem uporabil logaritemsko skalo.



Slika 3: Rezultati 500 permutacij

Na koncu sem pognal moj algoritem še za 500 permutacij, tokrat za alpho vrednost 0.05. Rezultata sta vidna na Sliki 3.

3.2 Hitrost izvajanja

10 permutacij:

```
real 8m20.437s
user 8m18.875s
sys 0m0.628s
```

100 permutacij:

```
real 73m46.716s
user 73m36.088s
sys 0m2.380s
```

500 permutacij:

```
real 377m52.873s
user 377m18.379s
sys 0m7.308s
```

Čas izvajanja sem želel pohitriti z boljšo izrabo dveh jeder mojega prenosnika. Za ta namen sem najprej napisal verzijo programa z nitmi. Uporabil sem funkcijo map iz razreda Pool, ki razdeli izvajanje funkcije v posamezne niti. Št. niti je določeno v konstruktorju. Ker pohitritve v delovanju nisem opazil, sem sklepal, da je problem, ker sta obe niti dostopali do seznama, kjer so se shranjevali rezultati.

Ker rešitev z nitmi ni delovala zadovoljivo, sem se odločil poskusiti še z dvema ločenima procesoma. Program sem spremenil tako, da je prvi proces računal vrednosti za lihe razrede, drugi pa za sode, vendar tudi tukaj nisem opazil nobene pohitritve. Še več, oba procesa s polovičnim naborom razredov sta se izvajala dalj časa, kot pa en proces s polnim naborom razredov. Skelpam, da je to posledica velike količine dostopov do RAMa, in malo starejše arhitekture procesorja (Core 2 Duo). Na Core iX procesorjih s HyperThreading tehnologijo bi morala dvoprocesna verzija programa delovati bolje.

4 Izjava o izdelavi domače naloge

Domačo nalogo in pripadajoče programe sem izdelal sam.