Comments on `bnclassify` package runtimes

Bojan Mihaljevic, Concha Bielza, Pedro Larranaga

2015-11-18

Contents

1 Prediction

`bnclassify` implements fast prediction for augmented naive Bayes models with complete data. On the car evaluation data set (see `vignette("introduction", package="bnclassify")`) it is roughly 100 times faster than prediction with the `gRain` (Højsgaard 2012) package.

```
library(bnclassify)
data(car)
nb <- lp(nb('class', car), car, smooth = 0)
gr <- as_grain(nb)
library(microbenchmark)
microbenchmark(bnclassify = predict(nb, car),
               gRain  = gRain::predict.grain(gr, 'class', newdata = car),
               times = 1)
#> Unit: milliseconds
#>         expr        min         lq       mean     median         uq
#>  bnclassify   40.28362   40.28362   40.28362   40.28362   40.28362
#>       gRain 2656.59152 2656.59152 2656.59152 2656.59152 2656.59152
#>         max neval
#>    40.28362     1
#>  2656.59152     1
```

2 Wrapper algorithms

The wrapper algorithms can be computationally intensive. The following are runtimes for `tan_hc` on a Windows 7, 2.80 GHz, 16 GB RAM machine.

```
microbenchmark(
  tan_hc = {set.seed(0); t <- b <- tan_hc('class', car, k = 10,
                                          epsilon = 0)},
  tan_hc5 = {set.seed(0); t <- b <- tan_hc('class', car, k = 5,
                                          epsilon = 0)},
  times = 1)
#> Unit: seconds
#>     expr      min       lq     mean   median       uq      max neval
#>   tan_hc 1.568964 1.568964 1.568964 1.568964 1.568964 1.568964     1
#>  tan_hc5 1.074946 1.074946 1.074946 1.074946 1.074946 1.074946     1
```

5-fold cross-validation should take roughly 5 times more than learning.

```
tan_hc5 <- tan_hc('class', car, k = 5, epsilon = 0)
tan_hc5 <- lp(tan_hc5, car, smooth = 1)
microbenchmark(tan_hc = {set.seed(0); cv(tan_hc5, car, k = 5)},
               times = 1)
#> Unit: seconds
#>    expr      min       lq     mean   median       uq      max neval
#>  tan_hc 5.052269 5.052269 5.052269 5.052269 5.052269 5.052269     1
```

With the Soybean data set, which has 36 features, and 562 instances after removing the incomplete ones, `tan_hc` takes about 80 seconds on the above mentioned Windows 7 machine.

```
library(mlbench)
data(Soybean)
soy_complete <- na.omit(Soybean)
dim(soy_complete)
microbenchmark(
  tan_hc = {set.seed(0); tan_hc('Class', soy_complete, k = 5,
                                epsilon = 0)},
  times = 1)
```

3 Incomplete data

`bnclassify` uses `gRain` to compute the class posterior of instances with missing values (`NA`s). Even with a single `NA` in a dataset, runtime degrades significantly.

```
nb <- bnc('nb', 'class', car, smooth = 1)
car_na <- car
car_na[1, 4] <- NA
microbenchmark(predict(nb, car),
               predict(nb, car_na),
               times = 1)
#> Unit: milliseconds
```

```
#>                 expr      min       lq     mean   median       uq      max
#>    predict(nb, car) 44.92007 44.92007 44.92007 44.92007 44.92007 44.92007
#>  predict(nb, car_na) 77.52296 77.52296 77.52296 77.52296 77.52296 77.52296
#>  neval
#>      1
#>      1
```

This is especially relevant for wrapper learners, which call prediction during learning. It is therefore probably not a bad idea to use wrappers with incomplete data sets, unless these are rather small.

References

Højsgaard, Søren. 2012. "Graphical Independence Networks with the `gRain` Package for `R`." *Journal of Statistical Software* 46 (10): 1–26.