WB01

Łukasz Ławniczak

9 października 2017

Opis danych

```
library(randomForest)
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:randomForest':
##
##     margin
```

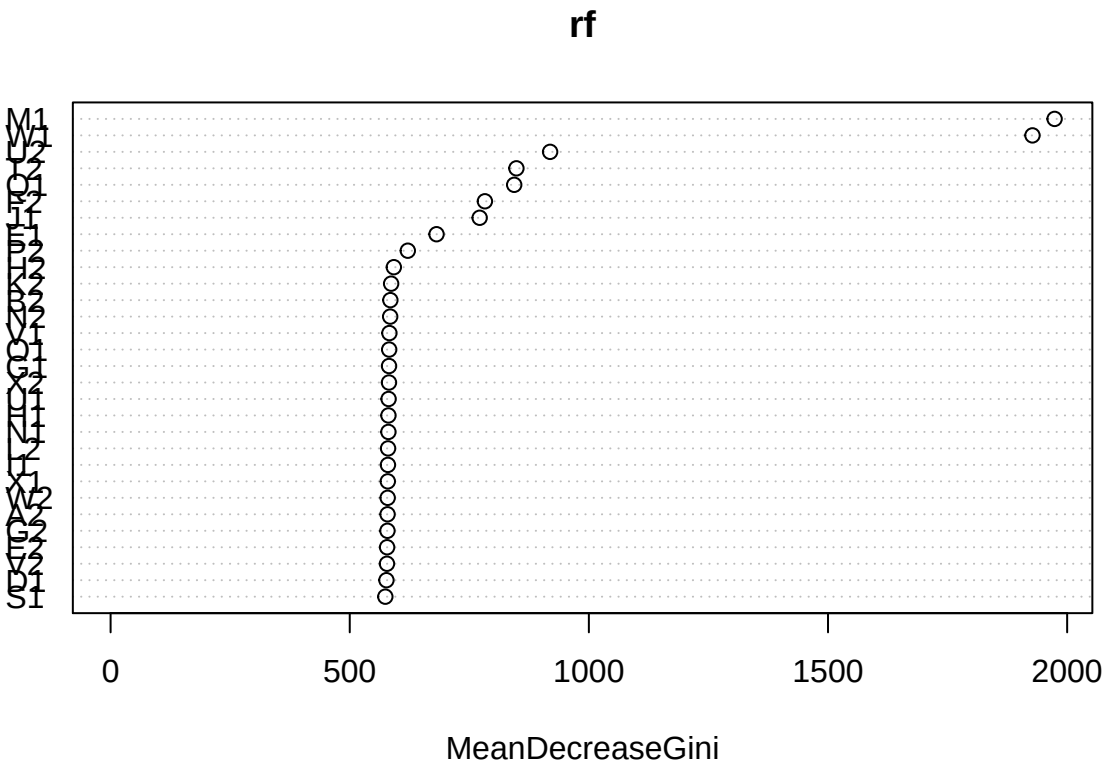
```
train <- read.table("zbior_uczacy.txt", h=TRUE, sep=";")
test <- read.table("zbior_testowy.txt", h=TRUE, sep=";")
```

Dane składają się ze zmiennych nominalnych przyjmujących 2 lub 4 wartości oraz zmiennych ciągłych. Wszystkie zmienne ciągłe przyjmują wartości z przedziału $[0, 100]$ i ich wartości są równomiernie rozmieszczone w tym przedziale. Zmienne nominalne również przyjmują wszystkie wartości z bardzo zbliżonym prawdopodobieństwem. Klasy zostały nazwane `klasa +` i `klasa -`. W zbiorze występuje podobna liczba próbek dla obu klas.

Selekcja zmiennych

Sprawdźmy istotne zmienne przy pomocy lasu losowego.

```
rf <- randomForest(y~., data=train)
varImpPlot(rf)
```



Istotne są zmienne M1, W1, U2, Q1, T2, F2, J1, E1, P2.

Zobaczmy jakie zmienne są istotne w modelu liniowym

```
lm <- glm(y~M1+W1+U2+Q1+T2+F2+J1+E1+P2, data=train, family=binomial(link="logit"))
summary(lm)
```

```
##
## Call:
## glm(formula = y ~ M1 + W1 + U2 + Q1 + T2 + F2 + J1 + E1 + P2,
##     family = binomial(link = "logit"), data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1085  -1.0713  -0.3502   1.0562   2.3182
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.3877036  0.0509464 -27.238  < 2e-16 ***
## M1          -0.0025635  0.0003301  -7.766 8.11e-15 ***
## W1           0.0135528  0.0003356  40.381  < 2e-16 ***
## U2          -0.0087921  0.0003310 -26.561  < 2e-16 ***
## Q1          -0.0083726  0.0003313 -25.274  < 2e-16 ***
## T2           0.0084070  0.0003337  25.192  < 2e-16 ***
## F2          -0.0001843  0.0003300  -0.559    0.576
## J1           0.0005906  0.0003295   1.793    0.073 .
## E1B          0.8474004  0.0271229  31.243  < 2e-16 ***
## E1C          0.8721584  0.0272674  31.985  < 2e-16 ***
```

```
## E1D          0.8434230  0.0272843  30.912  < 2e-16 ***
## P2B          0.8251283  0.0272255  30.307  < 2e-16 ***
## P2C          0.8172766  0.0271988  30.048  < 2e-16 ***
## P2D          0.8155740  0.0272406  29.940  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 69315  on 49999  degrees of freedom
## Residual deviance: 63092  on 49986  degrees of freedom
## AIC: 63120
##
## Number of Fisher Scoring iterations: 4
```

Spośród zmiennych wskazanych przez las losowy jedynie zmienne F2 i J1 wyszły nieistotne. Może to oznaczać, że występuje nieliniowa zależność między y a F2 i J1.

Wśród istotnych zmiennych występują dwie zmienne nominalne - E1 i P2. Rozkład y w zależności od E1 wygląda następująco:

```
table(train$y, train$E1)
```

```
##
##              A    B    C    D
##   klasa - 8078 5752 5577 5600
##   klasa + 4552 6891 6849 6701
```

Rozkład dla zmiennej P2 przedstawiono poniżej:

```
table(train$y, train$P2)
```

```
##
##              A    B    C    D
##   klasa - 7980 5690 5707 5630
##   klasa + 4613 6796 6807 6777
```

Prawdopodobieństwo klasy + przy wartościach B, C i D obu czynników jest bardzo zbliżone, więc informacja czy te czynniki przyjęły poziom A powinna być wystarczająca.

Utworzenie modelu

Model szacuje wartość `score` - prawdopodobieństwo, że próbka należy do klasy +. Jakością modelu jest częstotliwość występowania klasy + wśród 20% rekordów o najwyższej wartości `score`. W celu oceny jakości wykorzystano walidację prostą - podzielono zbiór treningowy na dwa równoliczne podzbiory. Przedstawiono jakość na zbiorze treningowym i testowym. Modele wykorzystują jedynie istotne zmienne (ograniczenie to nie wpływa znacząco na jakość modelu).

```
set.seed(999)
sel <- sample(nrow(train), 25000)
tr1 <- train[sel,]
te1 <- train[-sel,]
quality <- function(y, cls) {
  mean("klasa +" == head(cls[order(y, decreasing=TRUE)], 0.2*length(cls)))
}
```

Jakość lasów losowych wynosi ok. 0.83.

```
formula <- y~M1+W1+U2+Q1+T2+F2+J1+E1+P2
rf1 <- randomForest(formula, data=tr1, maxnodes=150)
y.tr <- predict(rf1, tr1, type="prob")[,2]
y.te <- predict(rf1, te1, type="prob")[,2]
c(train = quality(y.tr, tr1$y), test = quality(y.te, te1$y))
```

```
##  train   test
## 0.8634 0.8306
```

Model liniowy ma niższą jakość ze względu na to, że nie potrafi właściwie obsłużyć zmiennych F2 i J1.

```
lm1 <- glm(formula, data=tr1, family=binomial(link="logit"))
y.tr <- predict(lm1, tr1, type="response")
y.te <- predict(lm1, te1, type="response")
c(train=quality(y.tr, tr1$y), test=quality(y.te, te1$y))
```

```
##  train   test
## 0.7282 0.7310
```

XGBoost zachowuje się bardzo podobnie do lasu losowego.

```
library(xgboost)
```

```
## Warning: package 'xgboost' was built under R version 3.4.2
```

```
xgb <- xgboost(data=model.matrix(formula, tr1),
               label=as.numeric(tr1$y)-1,
               objective="binary:logistic", nrounds=15)
```

```
## [1]  train-error:0.329040
## [2]  train-error:0.310200
## [3]  train-error:0.303080
## [4]  train-error:0.294920
## [5]  train-error:0.285920
## [6]  train-error:0.274840
## [7]  train-error:0.266800
## [8]  train-error:0.261960
## [9]  train-error:0.259480
## [10] train-error:0.254480
## [11] train-error:0.249680
## [12] train-error:0.246720
## [13] train-error:0.244800
## [14] train-error:0.241800
## [15] train-error:0.240160
```

```
y.tr <- predict(xgb, model.matrix(formula, tr1))
y.te <- predict(xgb, model.matrix(formula, te1))
c(train=quality(y.tr, tr1$y), test=quality(y.te, te1$y))
```

```
##  train   test
## 0.8850 0.8368
```

Podsumowanie

Najskuteczniejszy okazał się model XGBoost i został on wykorzystany do obliczenia prawdopodobieństw na zbiorze testowym. Wyniki w pliku lukasz_lawniczak.txt.