WYKŁAD VI: Metody łączenia klasyfikatorów i estymatorów regresji: bagging i boosting.

MiNI PW, semestr letni 2013/2014

Czy jesteśmy w stanie poprawić jakość klasyfikatorów i estymatorów regresji przez ich łączenie? Często tak. Intuicja:

Rozwiązujemy problem klasyfikacji dwuklasowej przy użyciu C *niezależnych* klasyfikatorów. Ostateczna decyzja: większościowa, tzn. klasyfikujemy do populacji, którą wybrało najwięcej klasyfikatorów.

Załóżmy, że $P(\text{poprawna decyzja pojedyńczego klasyfikatora}) = 0.55$.
N- liczba poprawnych decyzji wśród C klasyfikacji.
N ma rozkład dwumianowy $Bin(C, 0.55)$.

Decyzja większościowa poprawna, gdy

$$N > 0.5C \iff N/C > 0.5.$$

$E(N/C) = 0.55$ i $\mathrm{Var}(N/C) = (0.45 \times 0.55)/C \to 0$ gdy $C \to \infty$. Zatem

$$P(N \leqslant 0.5C) \quad \text{małe dla dużych } C.$$

Obserwacja zwana często twierdzeniem Condorceta.

Bagging

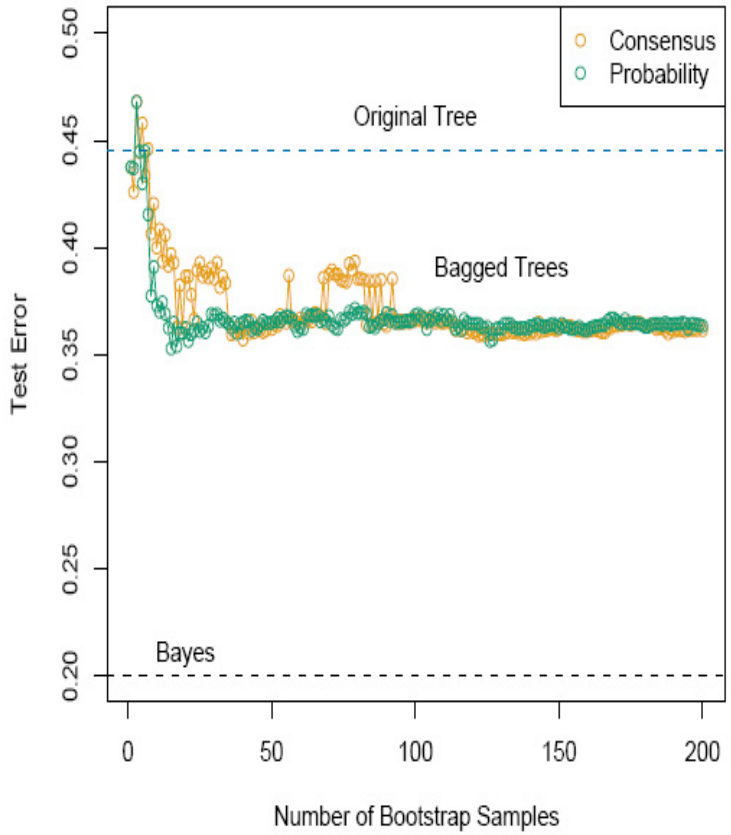
Tak jest często również dla przypadku zależnych klasyfikatorów.
Zjawisko 'zbiorowej mądrości' (collective wisdom).

Bagging (Bootstrap aggregating)

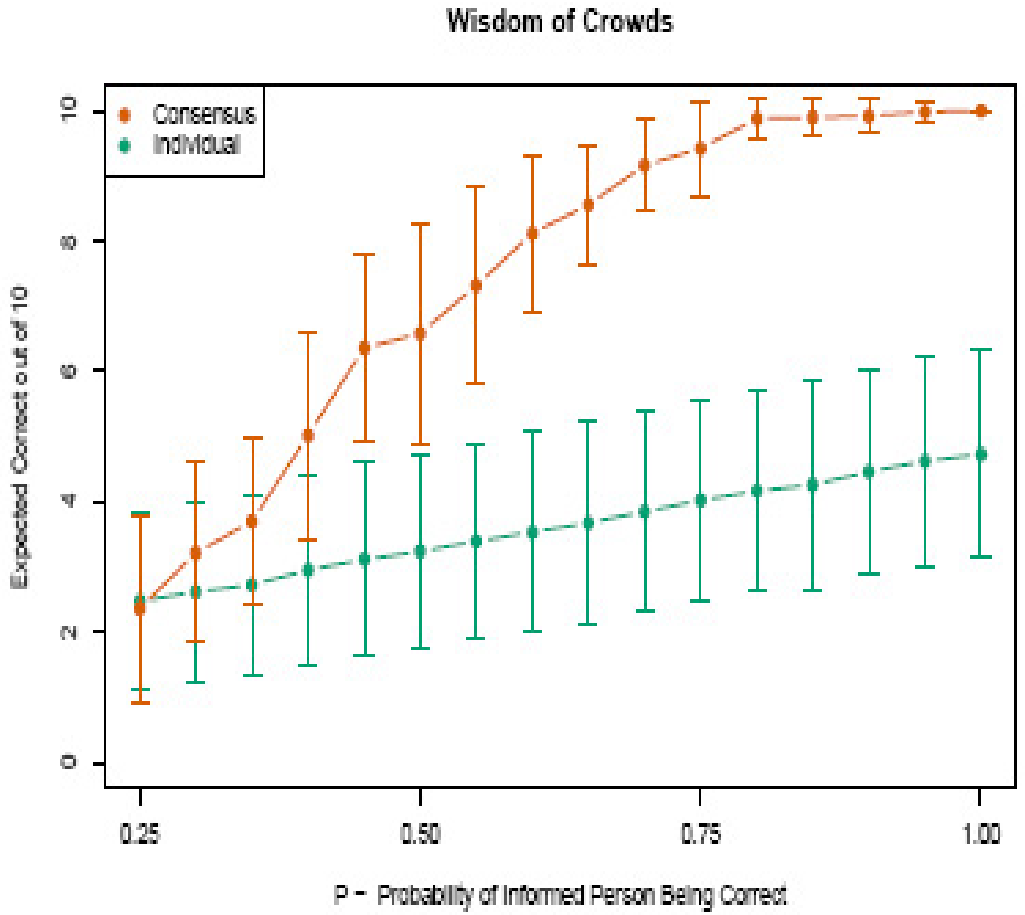
Dla ustalonej próby uczącej losujemy z niej B prób bootstrap $\mathcal{U}^{*1}, \ldots, \mathcal{U}^{*B}$. Na podstawie i-tej tworzymy klasyfikator $\hat{d}^{*i}(\mathbf{x})$ lub estymator regresji $\hat{f}^{*i}(\mathbf{x})$.
Dla problemu klasyfikacji: reguła większościowa;
Dla problemu estymacji regresji: $\hat{f}_{bag}(\mathbf{x}) = (1/B) \sum_{i=1}^{B} \hat{f}^{*i}(\mathbf{x})$.

Bagging: zmniejsza wariancję przez uśrednianie, zwiększa stabilność klasyfikatorów i estymatorów regresji. Wada: tracimy interpretowalność klasyfikatora. Klasyfikator otrzymany przez bagging drzew nie jest drzewem !

Przykład. $n = 30$, $p = 5$, $X_i \sim N(0, 1)$, $cor(X_i, X_j) = 0.95$, $P(Y = 1|X_1 > 0.5) = 0.8$. $P(Y = 1|X_1 \leqslant 0.5) = 0.2$. Klasyfikator: drzewo bez przycinania, 200 prób bootstrap, ocena na podstawie próby testowej 2000 elementów.



Przykład.(ESL, str. 287) 50 osób głosuje w 10 kategoriach, w każdej 4 kandydatów. 15 z 50 osób eksperci (z prawdopodobieństwem P wybiera prawdziwego kandydata), inne głosują losowo. Badana oczekiwana liczba nominowanych prawdziwych kandydatów dla pojedyńczego eksperta i całego zespołu.



Boosting (Freund, Schapire (1996))

Różnica w porównianiu z baggingiem: rozkład prawdopodobieństwa, według którego losuje się elementy do pseudpróby zmienia się z pseudopróby na pseudopróbę.
Wagi początkowe jak w baggingu: $w_i = 1/n$, $i = 1, 2, \ldots, n$. Jeśli pewien klasyfikator, o numerze C, źle sklasyfikował element i-ty, to klasyfikator $C + 1$ zwiększy wagę w_i przypisaną temu elementowi.

Jak uwzględniać wagi w klasyfikatorze ?

Metoda ogólna: bootstrap ważony. Zamiast losować elementy do pseudopróby z prawdopodobieństwem $1/n$ każdy, losujemy i-ty element z prawdopodobieństwem w_i ($(w_1, w_2, \ldots, w_n)$: zadany wektor prawdopodobieństw).

Dla pewnych klasyfikatorów: metoda bezpośrednia.
Drzewa ważone. W drzewie klasycznym, w węźle określonym przez warunek $\mathbf{x} \in R_m$ wkład poszczególnych elementów w węźle wynosił

$$\frac{1}{\sum_{j=1}^{n} I\{\mathbf{x}_j \in R_m\}.}$$

W drzewie ważonym wkład elementu i wynosi

$$\frac{w_i}{\sum_{j=1}^{n} w_j I\{\mathbf{x}_j \in R_m\}}.$$

Prawdopodobieństwo $P(Y = 1|\mathbf{x} \in R_m)$, że element klasy w węźle $\mathbf{x} \in R_m$ jest z klasy 1 oceniane przez

$$\frac{\sum_{\mathbf{x} \in R_m} w_i I\{y_i = 1\}}{\sum_{\mathbf{x} \in R_m} w_j}.$$

Modyfikacja miar zróżnicowania uwzględniająca wagowanie.

Algorytm AdaBoost.M1

Klasy kodowane jako -1 i 1 ($Y \in \{-1, 1\}$), dane klasyfikatory $f_1, f_2, \ldots, f_C$, **w szczególności** $f_i = f, i = 1, \ldots, C$.
Konstrukcja łączenia działania tych klasyfikatorów (boosting).

- Przyjmij wagi $w_i = 1/n, i = 1, \ldots, n$;
- Dla $c = 1, 2, \ldots, C$:
  (i) Wytrenuj klasyfikator f_c na danych treningowych ważonych wagami $(w_1, w_2, \ldots, w_n)$;
  (ii) Oblicz:

$$err_c = \sum_{i=1}^{n} w_i I\{Y_i \neq f_c(X_i)\}$$

  Jeśli $err_c = 0$ lub $err_c \geqslant 0.5$ to przerwij. Jeśli nie, policz $\gamma_c = \log \frac{1-err_c}{err_c}$.

  (iii) $w_i \longleftarrow w_i \exp(\gamma_c I\{Y_i \neq f_c(X_i)\}, i = 1, 2, \ldots, n$

  (iv) znormalizuj wagi (w_i): $w_i := w_i / \sum_{i=1}^{n} w_i$.

Klasyfikator wynikowy oparty na C wytrenowanych klasyfikatorach:

$$f(x) := \text{sign}[\sum_{c=1}^{C} \gamma_c f_c(x)]$$

(i) Adaptacja wag odbywa się w ten sposób, żeby uczynić problem możliwie trudnym dla następnego klasyfikatora.

(ii) Klasyfikator z mniejszym err_c i co za tym idzie z większym γ_c ma większy wpływ na ostateczną decyzję (klasyfikator wynikowy). Inaczej niż w zwykłej regule większościowej !

(iii) Dla deterministycznej metody wyboru wag (jak np. dla drzew): AdaBoost jest algorytmem deterministycznym.

Dlaczego boosting (z tak określonymi wagami) działa ?

Statystyczne spojrzenie na boosting (Friedman, Hastie, Tibshirani (2000)).
AdaBoost jest metodą **sekwencyjnego** poszukiwania minimum funkcji

$$n^{-1}\sum_{i=1}^{n} L(Y_i, f(X_i)) \qquad L(y, f) = \exp(-yf) \qquad (*)$$

w klasie funkcji $f(x) = \sum_{c=1}^{C} \gamma_c f_c(x)$:

$$f_m(x) = f_{m-1}(x) + \gamma_m f_m(x), m = 1, \ldots, C, f_0 \equiv 0$$

Funcja straty $L(y, f) = \exp(-yf)$ ma następującą własność:

$$\min_f E(L(Y, f(X))|X = x) = \frac{1}{2}\frac{P(Y = 1|X = x)}{P(Y = -1|X = x)}$$

$2^{-1}P(Y = 1|X = x)/P(Y = -1|X = x)$ prowadzi do optymalnego klasyfikatora (bayesowskiego). Zatem minimalizując (*) znajdujemy jego przybliżenie.

Alternatywy dla AdaBoost

Inne funkcje straty mające podobne własności co $L(y, f) = \exp(-yf)$, w szczególności funkcja związana z funkcją wiarogodności

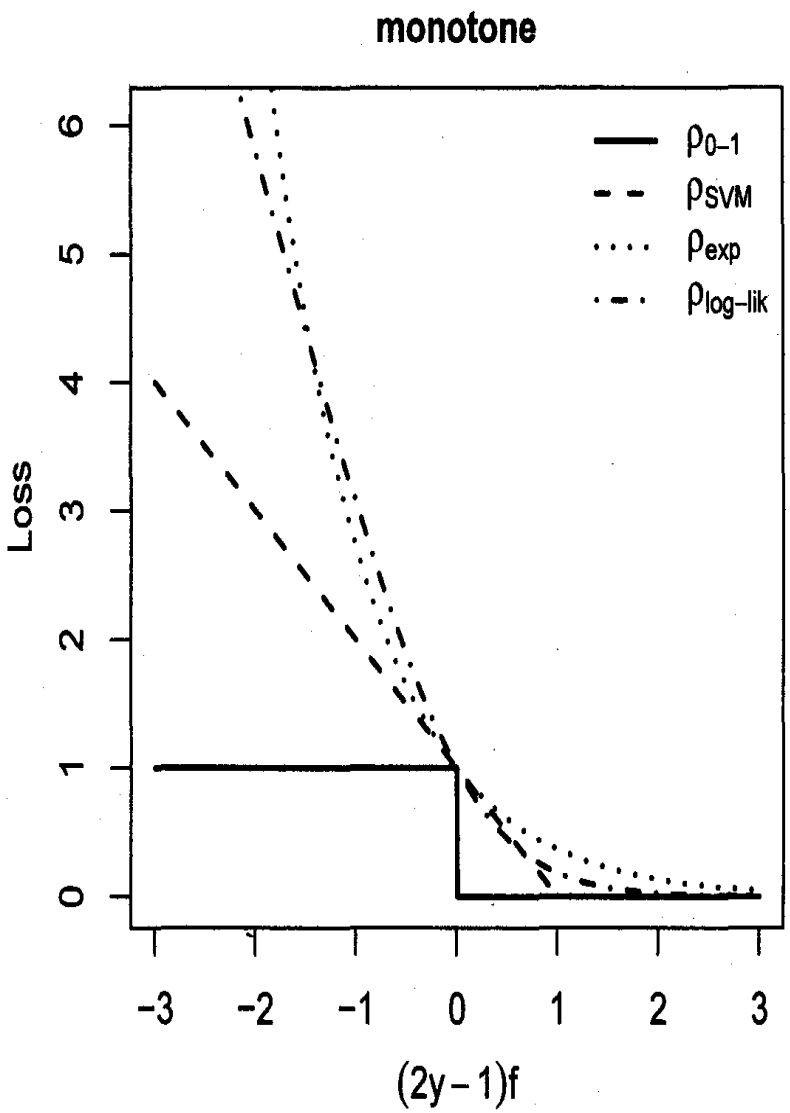
$$\tilde{L}(y, f) = \log_2(1 + \exp(-2fy)).$$

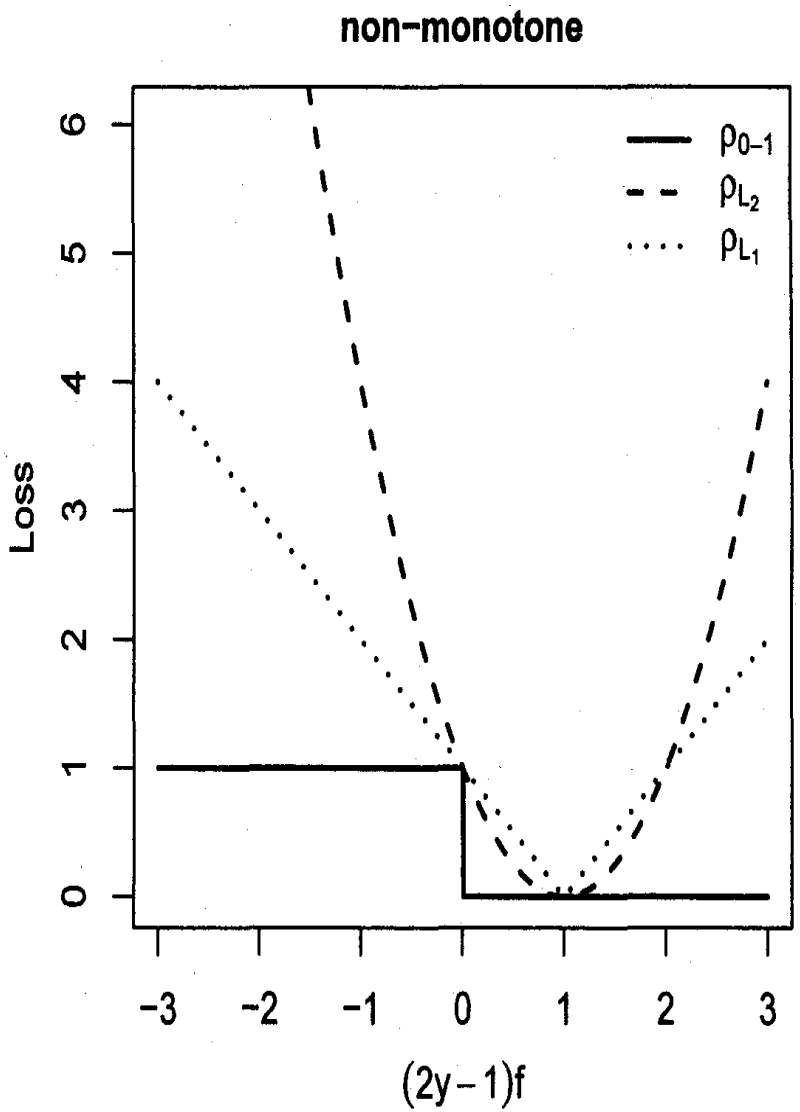
Prowadzi do algorytmu **BB (BinomialBoosting)**
Funkcja straty L^2:

$$\tilde{L}(y, f) = |y - f|^2 = |1 - yf|^2.$$

Prowadzi do algorytmu **L2boosting.**

$$L_{Bayes}(y, f) = I\{yf \leqslant 0\} \qquad L_{SVM} = |1 - yf|_+ \ldots$$


monotone
Loss
(2y − 1)f
ρ0−1
ρSVM
ρexp
ρlog−lik
0
1
2
3
4
5
6
−3
−2
−1
0
1
2
3


non−monotone
Loss
(2y − 1)f
ρ0−1
ρL2
ρL1
0
1
2
3
4
5
6
−3
−2
−1
0
1
2
3

Arcing Adaptive Resampling and Combinining (Breiman)

Algorytm Arc-x4

- Przyjmij wagi $w_i = 1/n, i = 1, \ldots, n$;
- Dla $c = 1, 2, \ldots, C$:
  (i) Wytrenuj klasyfikator f_c na danych treningowych ważonych wagami $(w_1, w_2, \ldots, w_n)$;
  (ii) Oblicz:
  frakcję p_i klasyfikatorów $f_1, f_2, \ldots, f_c$ błędnie klasyfikujących $(\mathbf{x}_i, y_i)$.
  (iii) $w_i \longleftarrow \frac{1+p_i^4}{\sum_{k=1}^{n} 1+p_k^4}$, $i = 1, 2, \ldots, n$

Klasyfikacja większościowa na podstawie wytrenowanych klasyfikatorów $f_1, \ldots, f_C$.
Arc-x4 działa porównywalnie do AdaBoost, często lepiej dla małych zbiorów danych.

Funkcyjny algorytm gradientowy FAG

Ogólna metoda zaproponowana przez Friedmana, zarówno dla modeli klasyfikacyjnych jak i regresyjnych.
Szukamy funkcji minimalizujacej ryzyko empiryczne:
$\operatorname{argmin}_f n^{-1} \sum_{i=1}^{n} L((Y_i), f(X_i))$

- Inicjalizacja: $\hat{f}^{[0]}(\cdot) \equiv \operatorname{argmin}_c n^{-1} \sum_{i=1}^{n} L(Y_i, c)$.
  Dla $m = 1, \ldots, m_{stop}$:
- (i) Oblicz

$$U_i = -\frac{\partial}{\partial f} L(Y_i, f)_{|f=\hat{f}^{[m-1]}(X_i)} \quad i = 1, 2, \ldots, n.$$

(ii) Zastosuj wybraną metodę oszacowania funkcji regresji do próby (X_i, U_i):

$$(X_i, U_i) \longrightarrow \hat{g}^{[m]}(\cdot)$$

(szacowanie gradientu).
(iii) $f^{[m]}(\cdot) = f^{[m-1]}(\cdot) + \nu \times \hat{g}^{[m]}(\cdot)$

Dla C drzew regresyjnych algorytm gradientowy ma postać:

- $\hat{f}^{[0]}(\cdot) = \bar{y}$
  Dla $m = 1, \ldots, m_{stop}$:
- $u_i = y_i - f^{[m-1]}(\mathbf{x}_i)$;
- Dopasuj drzewo regresyjne do danych $(\mathbf{x}_i, u_i)$ uzyskując rozbicie na hiperkostki R_{jm}, $j = 1, \ldots, J_m$;
- Dla $j = 1, \ldots, J_m$ oblicz

$$a_{jm} = \operatorname{argmin} \sum_{\mathbf{x}_i \in R_{jm}} (u_i - a_{jm})^2 = \frac{\sum_{\mathbf{x}_i \in R_{jm}} u_i}{|R_{jm}|}.$$

$$f^{[m]}(\mathbf{x}_i) = f^{[m-1]}(\mathbf{x}_i) + \sum_{i=1}^{J_m} a_{jm} I\{\mathbf{x}_i \in R_{jm}\}$$

pakiety `gbm`, `mboost`.

Przykład. Stosujemy metodę AdaBooost do podzbioru `vehicleOS` danych `vehicle` składającego się z typów van i saab (funkcja `boost` z biblioteki `adaboost`). Próba losowo podzielona na treningową ($n_1 = 250$ i testową $n_2 = 166$.

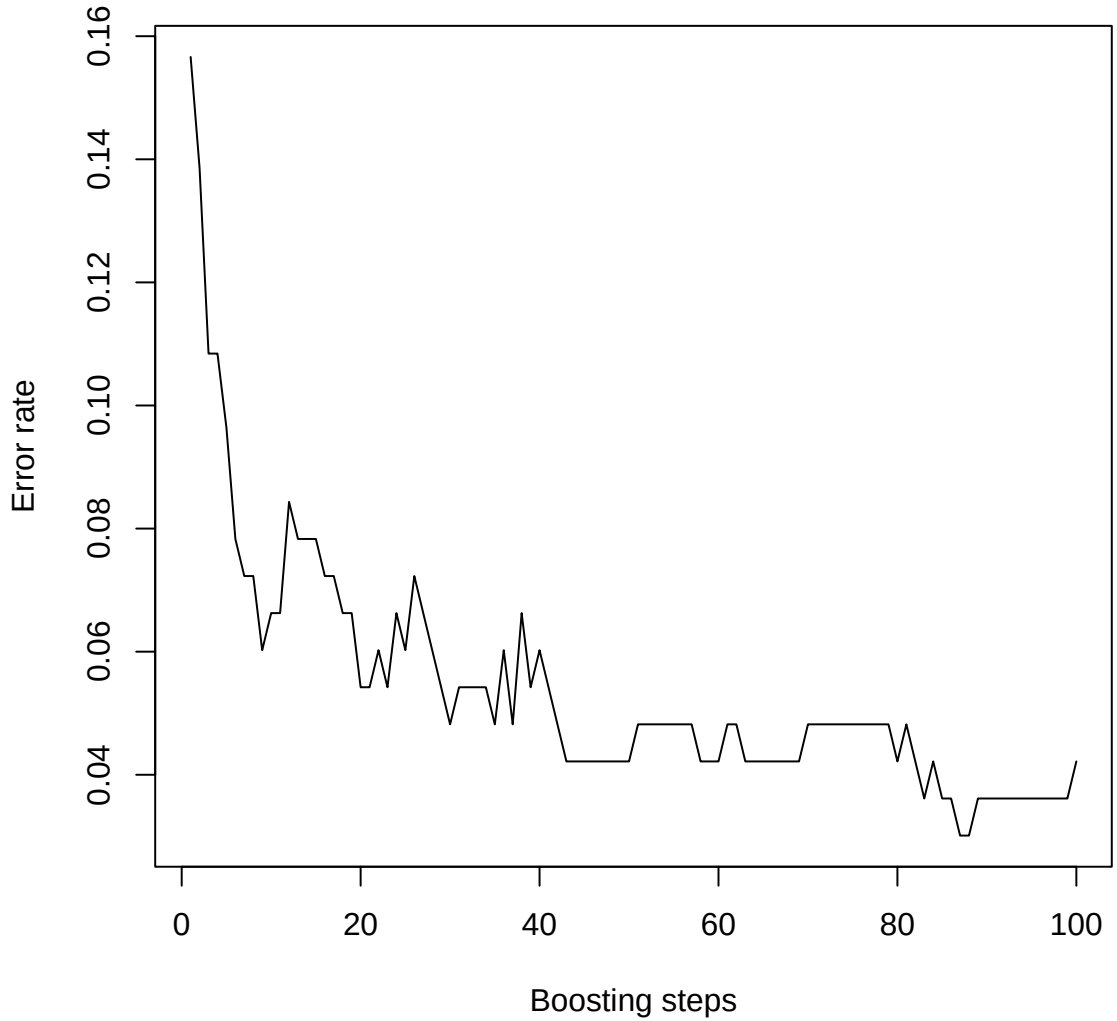
```
losnum <- sample(1:length(vehicleOS[ ,1]), replace=FALSE)
traintest <- data.frame(vehicleOS[losnum,1:18],
Y=factor(vehicleOS[losnum,19]))
n1 <- 250, n2<-166

xlearn <- as.matrix(traintest[c(1:n1), 1:18])
ylearn <- c(ifelse(traintest[c(1:n1), 19]=="saab",0,1))
xtest  <- as.matrix(traintest[(n1+1):n12, 1:18])
ytest  <- c(ifelse(traintest[(n1+1):n12, 19]=="saab",0,1))
```

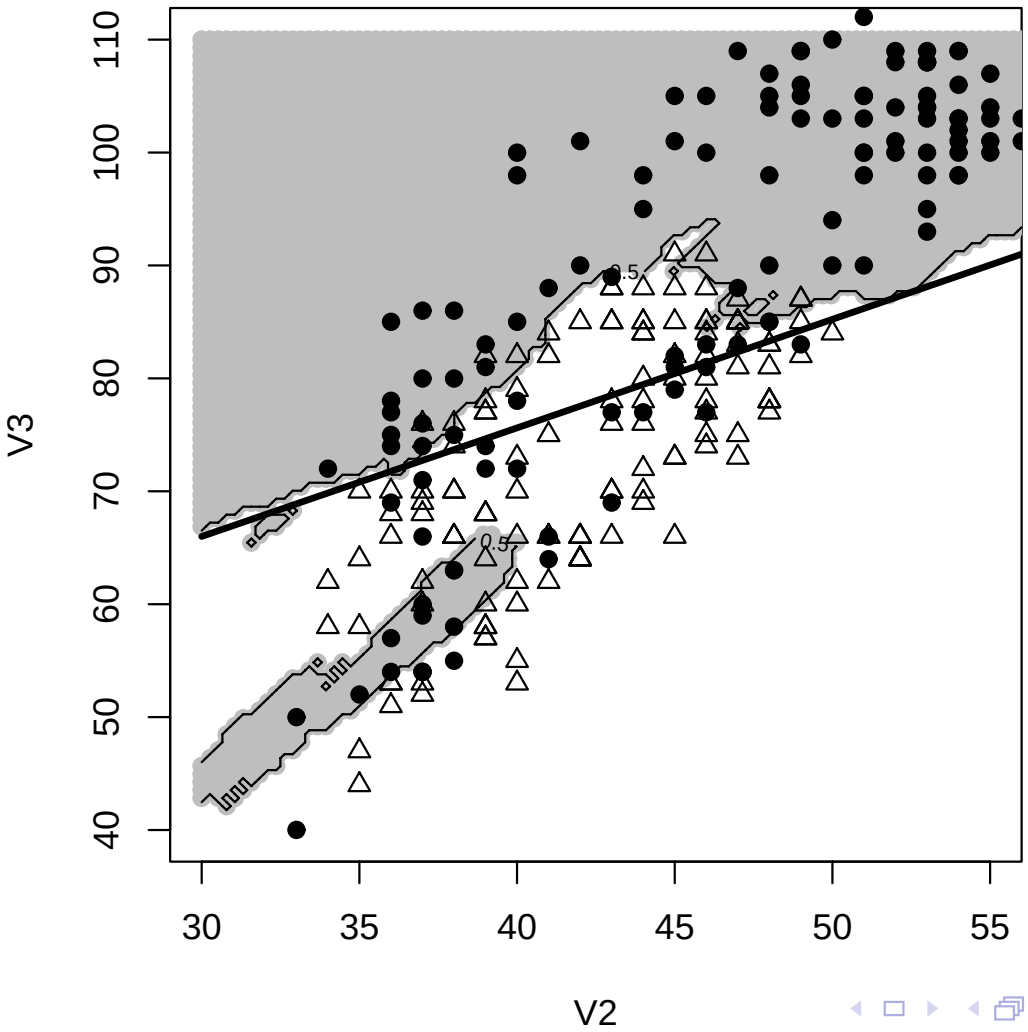
(konieczna zamiana wartości "saab" i "van" na 0 i 1.) Podstawowy klasyfikator: drzewo klasyfikacyjne. Funkcja `adaboost` wywołuje funkcję `learner`, która wyznacza drzewo w oparciu o `rpart`. Ustawienie parametrów `rpart` w `cntrl`.

```
cntrl <- rpart.control(maxdepth = 6, minsplit = 5,
        maxsurrogate = 0, usesurrogate = 0, maxcompete = 1, cp = 0,
        xval = 0)
 ..............
fit <- adaboost(xlearn, ylearn, xtest, presel=0, mfinal=100)
#C=100
kl <- ifelse(c(fit[ ,100])>0.5, 1, 0)
 tab4 <- print(table(ytest, kl))
    kl
ytest  0  1
    0 74  6
    1  1 85
 print(proc4 <- sum(diag(tab4))/sum(tab4))
[1] 0.9578
```

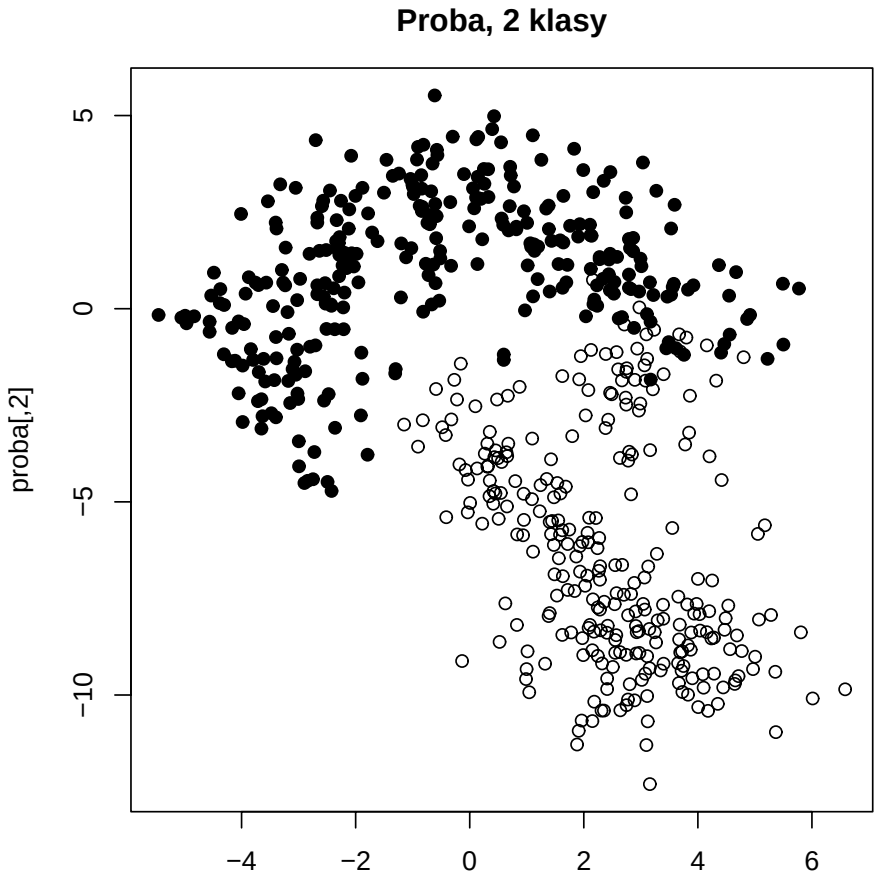
Estymator błędu klasyfikacji 0.042. Dla pojedyńczego drzewa (z cp=0.0001) dla próby testowej otrzymujemy błąd równy 0.1386.

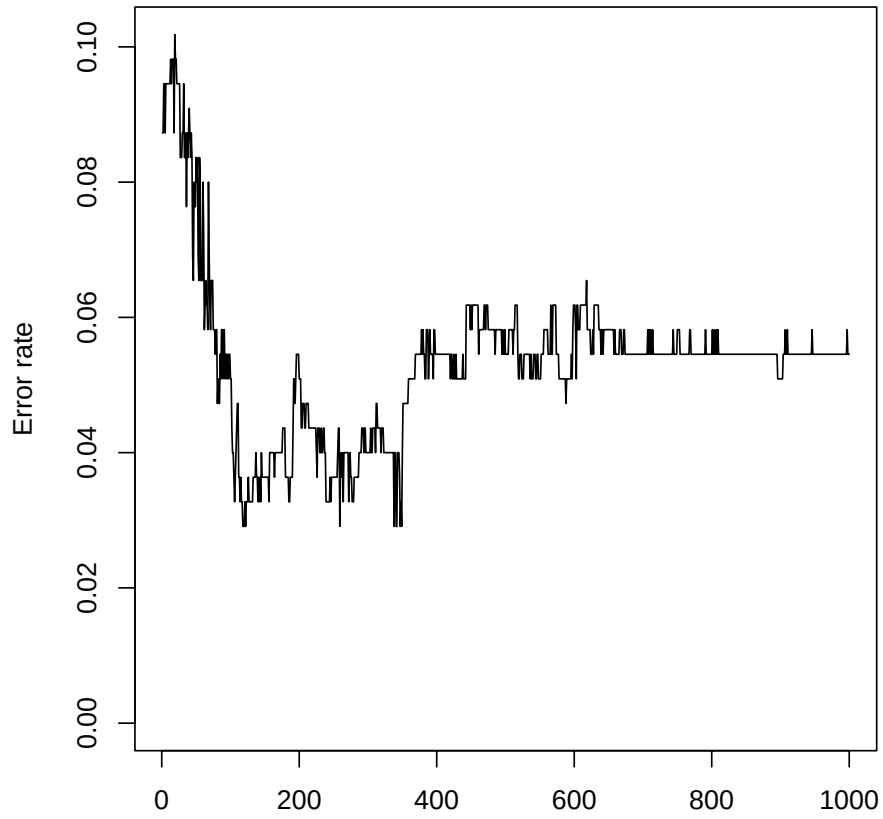

Error rate
0.04
0.06
0.08
0.10
0.12
0.14
0.16
0
20
40
60
80
100
Boosting steps

AdaBoost zastosowana do LDA (ten sam zbiór danych, tylko dwa atrybuty: V2 i V3). $C = 200$



Istotny problem w boostingu: wybór liczby klasyfikatorów (liczby iteracji). Błąd nie musi maleć wraz z liczbą iteracji! Przetrenowanie!





Lasy Losowe

Wersja Forest-RI Breimana (1999).
Tak jak w algorytmie bagging, losujemy kolejne próby bootstrap o liczności n i na ich podstawie budujemy drzewa (bez przycinania).
W każdym węźle budowanego drzewa losujemy (bez zwracania) podzbiór m spośród p atrybutów i tylko wylosowane atrybuty są rozpatrywane przy wyborze reguly podzialu.
Losowania atrybutów dokonywane są niezależnie dla każdego węzła. Liczbę m wybiera się adaptacyjnie lub stosuje się $m = \sqrt{p}$. Klasyfikacja nowych obserwacji: większość wskazań skonstruowanych drzew zastosowanych do tej obserwacji.
Błąd klasyfikacji takiego lasu losowego ocenia się stosując kolejno do n elementów próby wszystkie klasyfikatory, przy konstrukcji których te elementy *nie* były wykorzystywane. Ponieważ próba bootstrap zawiera przeciętnie $2/3 \times n$ różnych elementów, głosowanie na podstawie średnio $1/3$ ogólnej liczby wszystkich drzew, do konstrukcji których ten element nie był użyty.

W ten sposób otrzymamy ułamek błędnych klasyfikacji dla wszystkich elementów: estymator prawdopodobieństwa błędu klasyfikacji.
W metodzie lasów losowych możemy oceniać istotność zmiennych. Mianowicie, dla ustalonego drzewa i zmiennej m odejmujemy od liczby elementów próby uczącej poprawnie klasyfikowanych przez to drzewo liczbę elementów poprawnie sklasyfikowanych zmodyfikowanej próby. Modyfikacja polega na tym, że wartości zmiennej m zostały losowo przestawione (jeśli zmienna m nie występowała w drzewie, to różnica $= 0$).
Indeks istotności zmiennej to np. średnia wartość tak policzonych różnic dla wszystkich drzew w lesie.