Kinds of Ensembles

Tested on apple quality dataset

David Nardi

June 11, 2024

MSc in AI, University of Florence

Apple quality dataset

Variables

- `Size`
- `Weight`
- `Sweetness`
- `Crunchiness`
- `Juiciness`
- `Ripeness`
- `Acidity`

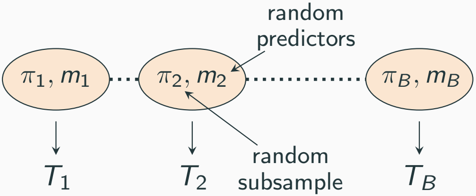
Binary classification task

Class distribution: 0.49 – 0.51

Methods

- kNN, Decision tree, Logistic regression
- Random forest
- AdaBoost
- Super learner

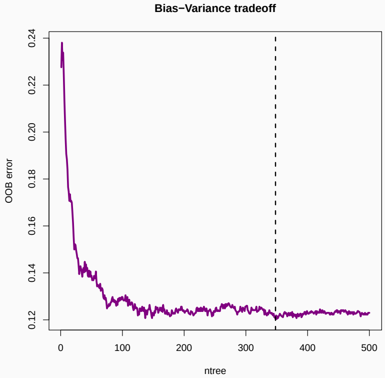
Random forest



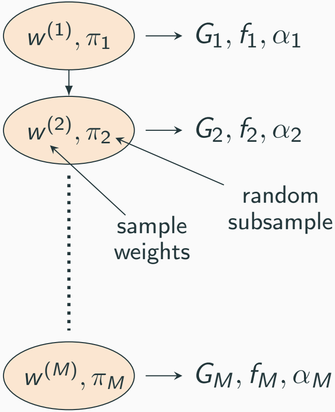

```
randomForest(x, y,
  importance=TRUE,
  ntree=B.oob ← B*
  )
```

$$G(x) = \arg\max_k \sum_{b=1}^{B} \mathbb{I}(T_b(x) = k)$$

$$\texttt{mtry} = \lfloor \sqrt{p} \rfloor$$



AdaBoost algorithm



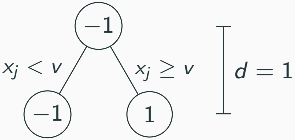
$$f_m(x) = f_{m-1}(x) + \lambda \alpha_m G_m(x)$$

$$G(x) = \text{sign}(f_M(x))$$

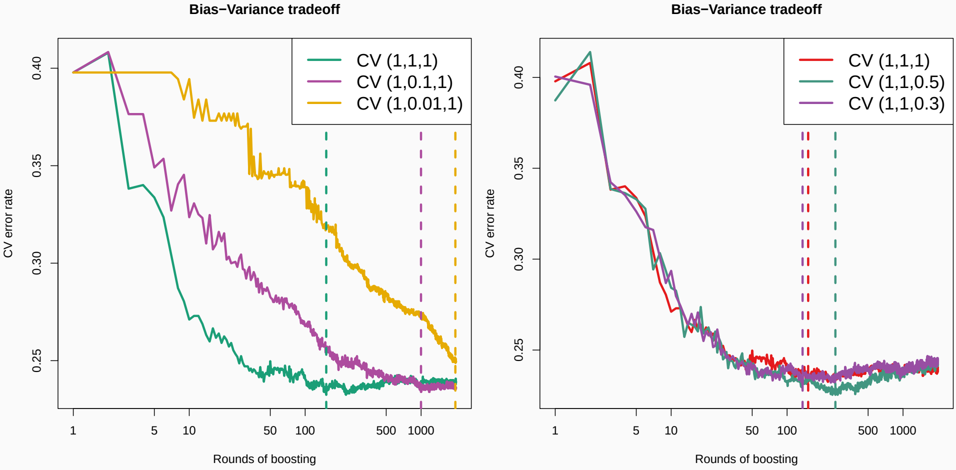
$$L(y, f(x)) = \exp(-yf(x))$$

Encoding $\mathcal{Y} \in \{-1, 1\}$

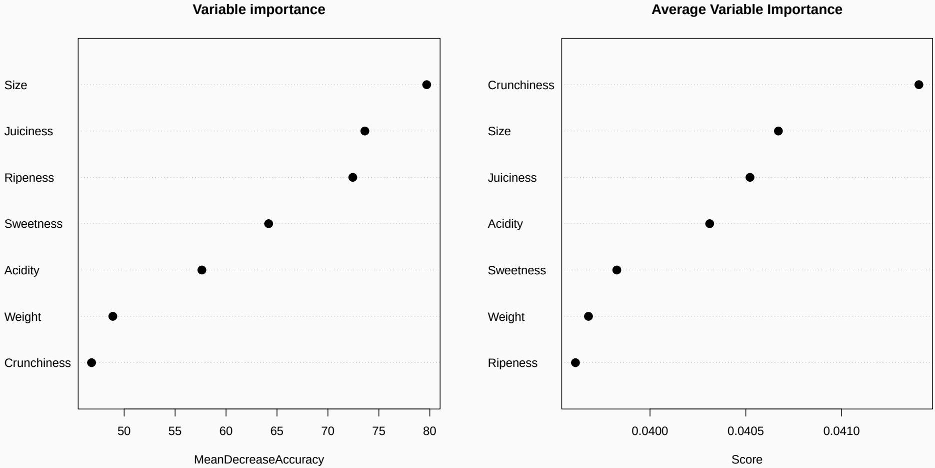
```
ada::ada(x, y,
  loss="exponential",
  type="discrete",
  iter ← M*, nu ← λ*,
  bag.frac ← π*,
  control=base.learner)
```



AdaBoost tuning



RF and Ada comparison



The Super Learner flow diagram

$$\hat{\alpha} = \arg\min_{\alpha} \sum_{i=1}^{N} L(Y_i, m(z_i|\alpha))$$

$$m(z|\alpha) = \sum_{k=1}^{K} \alpha_k \hat{\varphi}_{k,T(\nu)}(\boldsymbol{X}_{V(\nu)})$$

strong learners
on full $\mathcal{D}$, $\hat{\varphi}_k(\boldsymbol{X})$

$$\boxed{\hat{\varphi}_1, \hat{\varphi}_2, \dots, \hat{\varphi}_K}$$

learners
prediction

$$\mathcal{D} = \{(x_i, y_i)\}_1^N$$

$$\hat{\varphi}_{\mathrm{SL}} = \sum_{k=1}^{K} \hat{\alpha}_k \hat{\varphi}_k$$

prediction
weights

cross-validation
$\hat{\varphi}_{k,T(\nu)}(\boldsymbol{X}_{V(\nu)})$

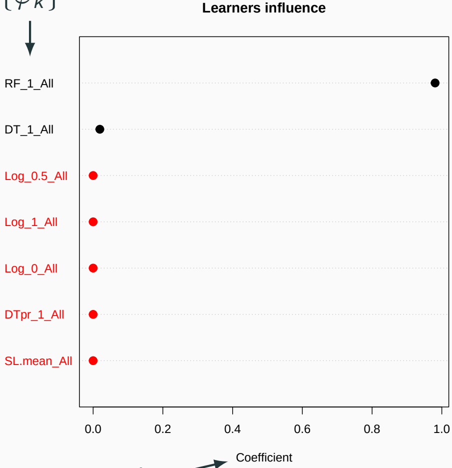
$\hat{\varphi}_1$	$\hat{\varphi}_2$	$\dots$	$\hat{\varphi}_K$	Y
1	1	$\dots$	1	1
2	2	$\dots$	2	2
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
V	V	$\dots$	V	V

Super learner in practice

```
SuperLearner(Y, X,
  cluster=cluster,
  SL.library=myLib,
  cvControl=list(
    V=10,shuffle=FALSE))
```

$\leftarrow \{\varphi_k\}$

$\{\hat{\varphi}_k\}$



$\hat{\alpha}_k$

Performance

Model	Train score	Test score
CART	0.0000	0.0000
Random forest	0.0000	0.0000
AdaBoost	0.0000	0.0000
Super learner	0.0000	0.0000

References i

T. Hastie, R. Tibshirani, and J. H. Friedman
The Elements of Statistical Learning
Springer, 2009.

E. C. Polley, and M. J. van der Laan
Super Learner in Prediction
U.C. Berkeley Division of Biostatistics Working Paper Series.
Working Paper 266, 2010

M. Culp, K. Johnson and G. Michailidis
`ada`: **The R Package Ada for Stochastic Boosting**
Journal of Statistical Software, 17(2), 1–27, 2006

Discrete AdaBoost algorithm

Discrete AdaBoost with shrinkage and out-of-bag, as an additive model with prediction function $f_m(x)$

Input: M, $\{(x_i, y_i)\}_1^N$, $x_i \in \mathbb{R}^p$

Initialize $f_0(x) = 0$;

for $m = 1$ **to** M **do**

  Set $w_i^{(m)} = -\frac{\partial L(y,g)}{\partial g}\Big|_{g=f_m(x)}$ s.t. $\sum_{i=1}^N w_i^{(m)} = 1$;

  Fit classifier $G_m(x)$ using $w_i^{(m)}$ with samples from π_m;

  Weighted error rate $\mathrm{err}_m = \sum_{i=1}^N w_i^{(m)} \mathbb{I}(y_i \neq G(x_i))$;

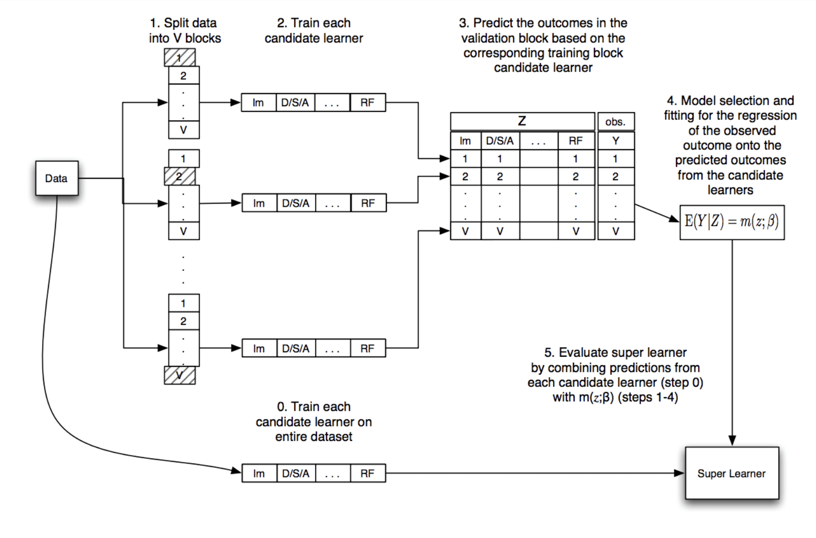
  Set $\alpha_m = \frac{1}{2} \log\left(\frac{1-\mathrm{err}_m}{\mathrm{err}_m}\right)$;

  Update $f_m(x) \leftarrow f_{m-1}(x) + \lambda \alpha_m G_m(x)$;

end

Output: $G(x) = \mathrm{sign}(f_M(x))$

Super learner algorithm flow diagram



Input: $\mathcal{D} = \{(x_i, y_i)\}_1^N$, $\mathcal{L} = \{\varphi_k(X)\}_{k=1}^K$

foreach *strong learner in* $\mathcal{L}$ **do**

  Fit φ_k on $\mathcal{D} \Rightarrow \hat{\varphi}_k(\boldsymbol{X}) \to \hat{\mathcal{L}} = \{\hat{\varphi}_k\}_{k=1}^K$;

end

for $\nu = 1, 2, \ldots, V$ **do**

  foreach *strong learner in* $\mathcal{L}$ **do**

    Fit φ_k on $T(\nu)$, predict $\hat{\varphi}_{k,T(\nu)}(X_i \in V(\nu))$;

  end

end

Stack output in an $N \times K$ matrix $Z = \left\{\hat{\varphi}_{k,T(\nu)}\left(X_{V(\nu)}\right)\right\}$;

Propose a family of weighted combinations

$$m(z|\alpha) = \sum_{k=1}^{K} \alpha_k \hat{\varphi}_{k,T(\nu)}\left(\boldsymbol{X}_{V(\nu)}\right) \to \hat{\alpha} = \arg\min_{\alpha} \sum_{i=1}^{N} L(Y_i, m(z_i|\alpha))$$

of size N s.t. $\alpha_k \geq 0$, $\sum_k \alpha_k = 1$ and minimizes $\sum_k \alpha_k \hat{\varphi}_k$;

Combine $\hat{\alpha}$ with the library $\hat{\mathcal{L}} \to \hat{\varphi}_{\text{SL}}(\boldsymbol{X}) = \sum_{k=1}^{K} \hat{\alpha}_k \hat{\varphi}_k(\boldsymbol{X})$;

Output: $\hat{\varphi}_{\text{SL}}$