

Statistical learning and prediction

Tree and ensemble methods

Get course files from <https://github.com/marjoleinF/IOPS-SLP>

Trees

Good: Easily interpretable and applicable

Bad: Not most accurate method

Ugly: Unstable



Example dataset

European Journal of Psychological Assessment, Vol. 17, Issue 2, pp. 130–136

Openness to Experience and Depression*

J.M. Carrillo, N. Rojo, M.L. Sánchez-Bernardos, and M.D. Avia

Departamento de Personalidad, Evaluación y Psicología Clínica, Facultad de Psicología,
Universidad Complutense de Madrid, Spain

Keywords: Openness to experience, depression, openness to fantasy, openness to actions, PB theory of depression, gender and depression, five factor model

Summary: The present study examines, in the context of the Five Factor Model, the contradictory role played by the *Openness to Fantasy* and *Openness to Actions* facets (of the *Openness to Experience* factor) in the prediction of depression. The fact that our data are taken from a sample of the Spanish general population is also a cross-cultural contribution that must be emphasized. 112 participants – 50% females and 50% males – filled out the NEO-PI and the BDI depression questionnaires. A stepwise regression shows that the *Fantasy* facet

Example dataset

Aim: predict depression based on personality scales

Predictor variables:

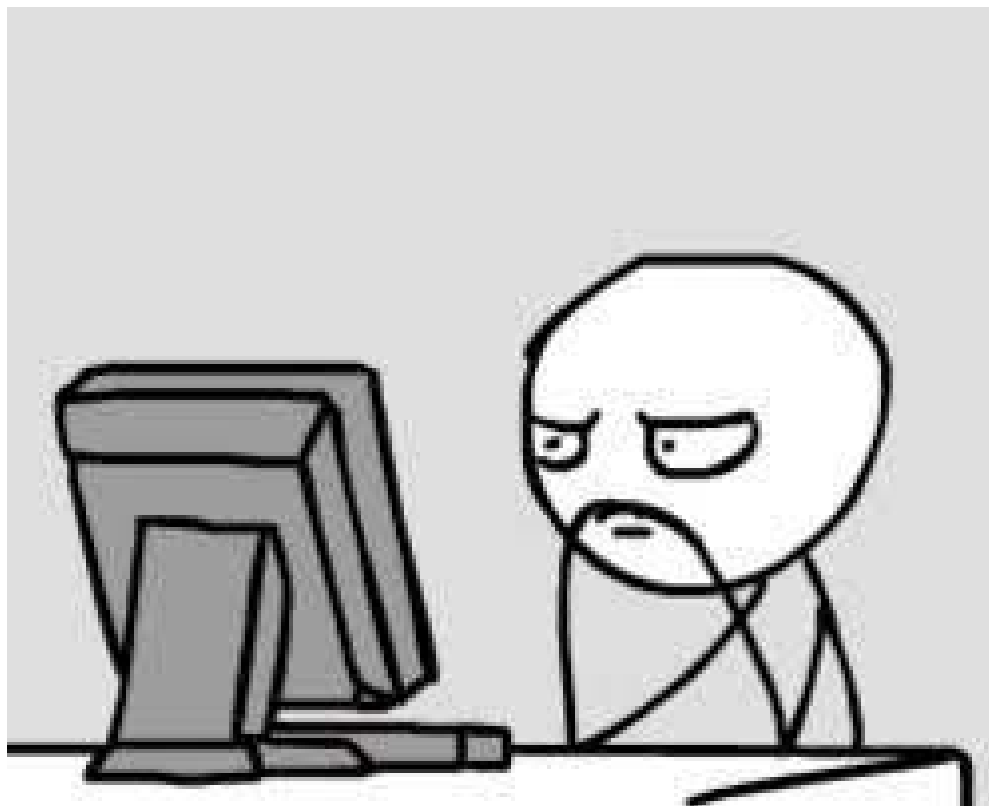
- Neuroticism: n1 through n6, ntot
- Extraversion: e1 through e6, etot
- Openness to experience: open1 through open5, opentot
- Altruism: altot
- Conscientiousness: contot
- Covariates: sex (sexo) and age (edad)

Response variable:

- Beck Depression Inventory total score (bdi)

Sample size: N = 112

R code example



Tree-growing algorithms

(e.g., CART)

- A decision tree divides $\{X_1, X_2, \dots, X_p\}$ into a set of distinct regions $\{R_1, R_2, \dots, R_J\}$, corresponding to the terminal nodes of the tree
- Regions should be as similar (pure) as possible with respect to the outcome Y
 - That is: $\text{var}(Y)$ should be as small as possible within each region
- Finding regions $\{R_1, R_2, \dots, R_J\}$ that provide a global optimum is computationally difficult (often infeasible)
- Most tree-growing algorithms turn it into a feasible task by employing:
 - Rectangular regions (i.e., each split defined by a single variable)
 - Only two-way splits
 - Greedy search (*current* best split is selected in each step, no looking ahead)
- Once tree is constructed, the mean of the outcome variable Y in region R_j gives \hat{Y} for new observations (sometimes, median may be preferred)

Splitting criteria

At each step, we seek variable j and value s that define

$$R_{\text{left}}(j,s) = \{ X \mid X_j < s \} \text{ and } R_{\text{right}}(j,s) = \{ X \mid X_j \geq s \},$$

which minimize the sum of the **loss function** in both regions

Different loss functions can be used, e.g.,

Regression trees:

$$\text{Absolute loss: } \sum_{i: x_i \in R_m} |y_i - \hat{y}_{R_m}|$$

$$\text{Squared error loss: } \sum_{i: x_i \in R_m} (y_i - \bar{y}_{R_m})^2$$

Classification trees:

$$\text{Classification error: } E = 1 - \max_k (\hat{p}_{mk})$$

$$\text{Gini index: } G = \sum_{k=1}^K \hat{p}_{mk} (1 - \hat{p}_{mk})$$

$$\text{Cross-entropy: } D = - \sum_{k=1}^K \hat{p}_{mk} \log \hat{p}_{mk}$$

Squared error loss and Gini index
are most often used

Both minimize the variance
within every terminal node

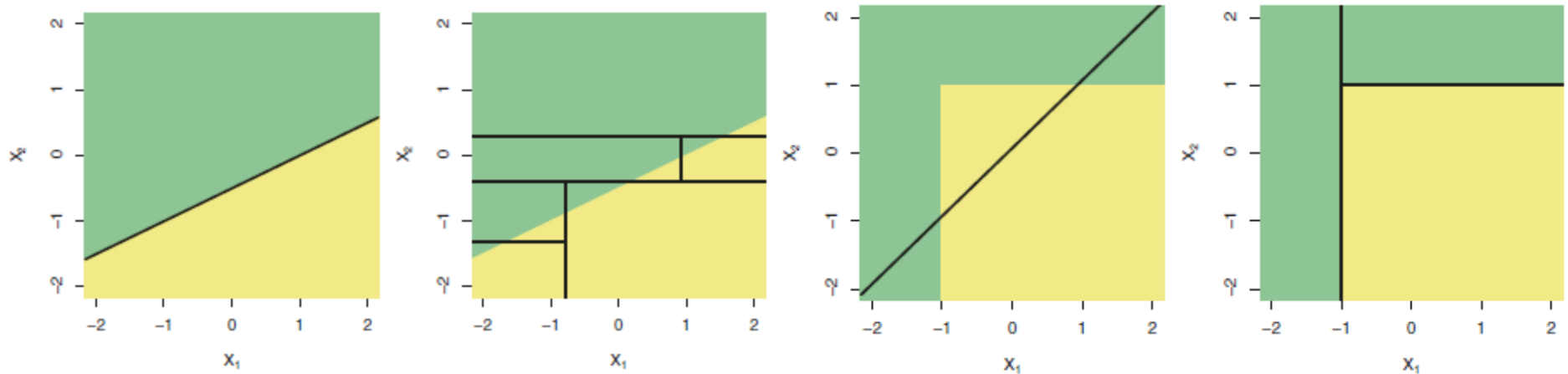
Exercise 1

Comparing misclassification indices

The bad

Trees do not provide optimal predictive accuracy, because:

- Small trees have low variance, but (likely) high bias
- Large trees have low bias, but high variance



The ugly - I: Instability

Trees have high variance / instability

- Each
- A split
varia
split.
the t
- Look
add c
and t
- How
very
Insta

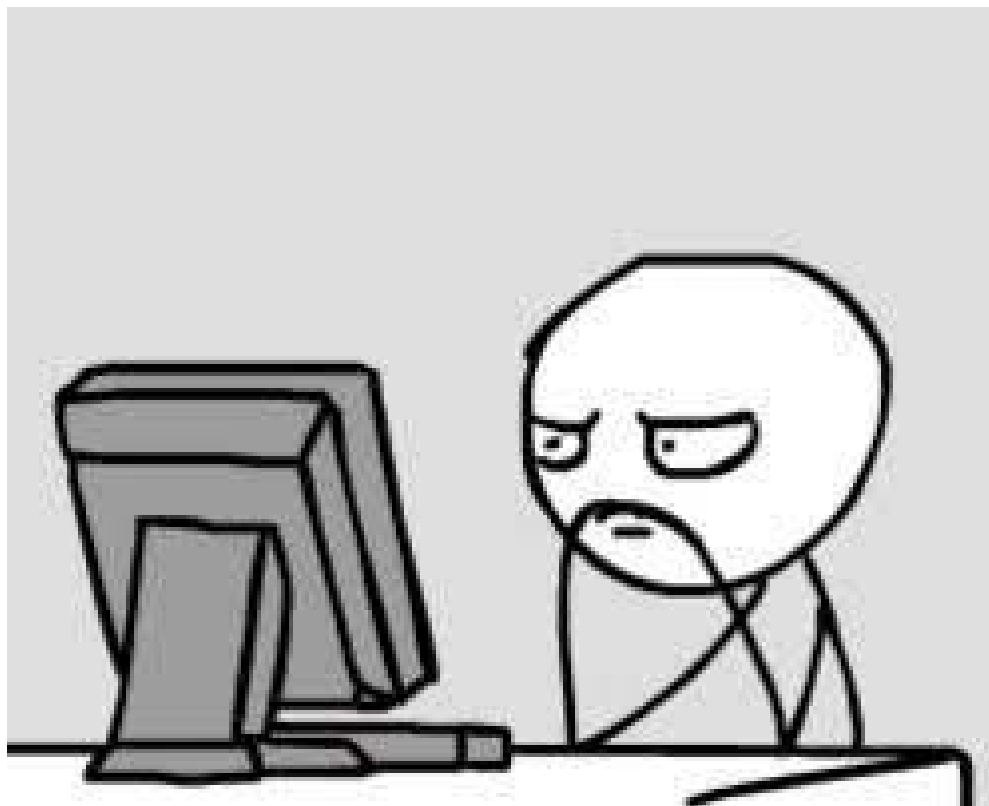


sampling
n that
ly affect

ire (e.g.,
e data

rovide
variates.

R code example



Balancing bias and variance

Solution 1: Pruning

tree should not be too small, nor too large

How?

1. grow full tree until impurity cannot be further reduced
2. cut off branches by selecting the tree that minimizes the penalized loss function:

$$\sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_j - \hat{y}_{R_m})^2 + \alpha |T|$$

where T equals the # of terminal nodes; α should be determined by k-fold CV

The ugly – II: Biased variable selection

CART performs exhaustive search: for every split, all possible splitting values are considered

-> biased variable selection: given two (or more) variables who are equally predictive of the outcome, variable with largest number of possible splitting values has higher probability of being selected for splitting

Solution: First select splitting variable, then select splitting value

-> unbiased variable selection: variables with equal predictive power have equal probability of being selected

Unbiased recursive partitioning

Conditional inference trees: Select splitting variable based on statistical tests

Separates variable and cutpoint selection

-> no selection bias towards variables with larger number of values

Provides Solution 2 for balancing bias and variance:
Yields natural stopping criterion: Significance level

Conditional inference trees

Statistical testing to select splitting variable:

H_0 : variable X_j and response Y are independent in the population

H_A : variable X_j and response Y are dependent in the population

Thus, for each variable X_j a p value under H_0 can be calculated, based on the observations in the current node

Variable X_j with lowest p value is selected for splitting the observations into two daughter nodes

If all p values in current node $>$ pre-specified value α , no further splitting is performed

After selecting splitting variable, select cutpoint as with CART (i.e., optimize loss function in daughter nodes)

Implemented in R in `ctree()` function of package 'partykit'

Exercise 2

Variable selection bias

Trees

Good: Easily interpretable and applicable

Bad: Not most accurate method

Ugly: Unstable



Tree ensembles

Single trees

- ▶ Small trees have low variance, but high bias
- ▶ Large trees have low bias, but high variance

Tree ensembles

- ▶ By combining the fitted (predicted) values from a large number of single trees, variance is reduced
- ▶ A price is paid in interpretability: instead of a single tree, we get a large number of trees

Ensemble learning

- ▶ Predictions of a large number of *weak learners* (e.g., single trees) are combined to create a *strong learner* (e.g., ensemble of trees)
 - ▶ Other learners than trees may be ensembled, too
- ▶ A weak learner is a method that has at least some predictive ability, that is at least better than random guessing
- ▶ Predictive accuracy of the full ensemble is always equal or better than that of any of its constituent members

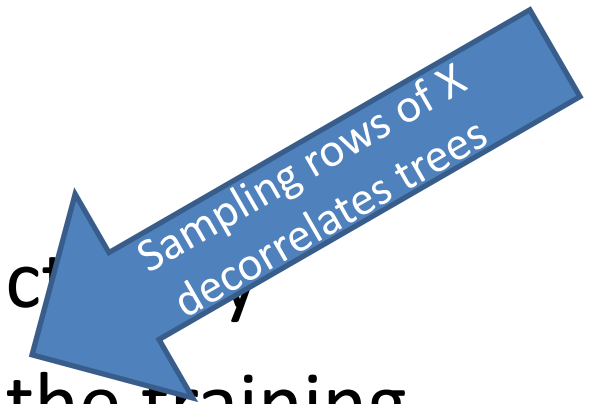
Ensemble learning

- ▶ Ensembling works well for unstable learning algorithms
(algorithms whose output undergoes major changes in response to small changes in the training data)
 - ▶ Unstable learning algorithms (generally, relatively): e.g., decision trees, neural networks, rule learning algorithms, OLS regression
 - ▶ Stable learning algorithms (generally, relatively): e.g., penalized linear regression, nearest neighbour, linear threshold algorithms
- ▶ Popular tree ensemble methods :
 - ▶ Bootstrap aggregation (bagging)
 - ▶ Random forests
 - ▶ Boosting

Bagging

A bagged tree ensemble is constructed

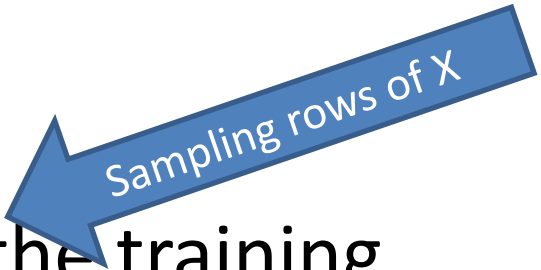
1. Taking bootstrap samples from the training data
2. Fitting a tree on each of the bootstrap samples
3. Combining the trees into an ensemble



Random forests

A random forest is constructed by

1. Taking bootstrap samples from the training dataset
2. Fitting a tree on each of the bootstrap samples, using a random subset of predictor variables for each split
3. Combining the trees into an ensemble



Sampling rows of X



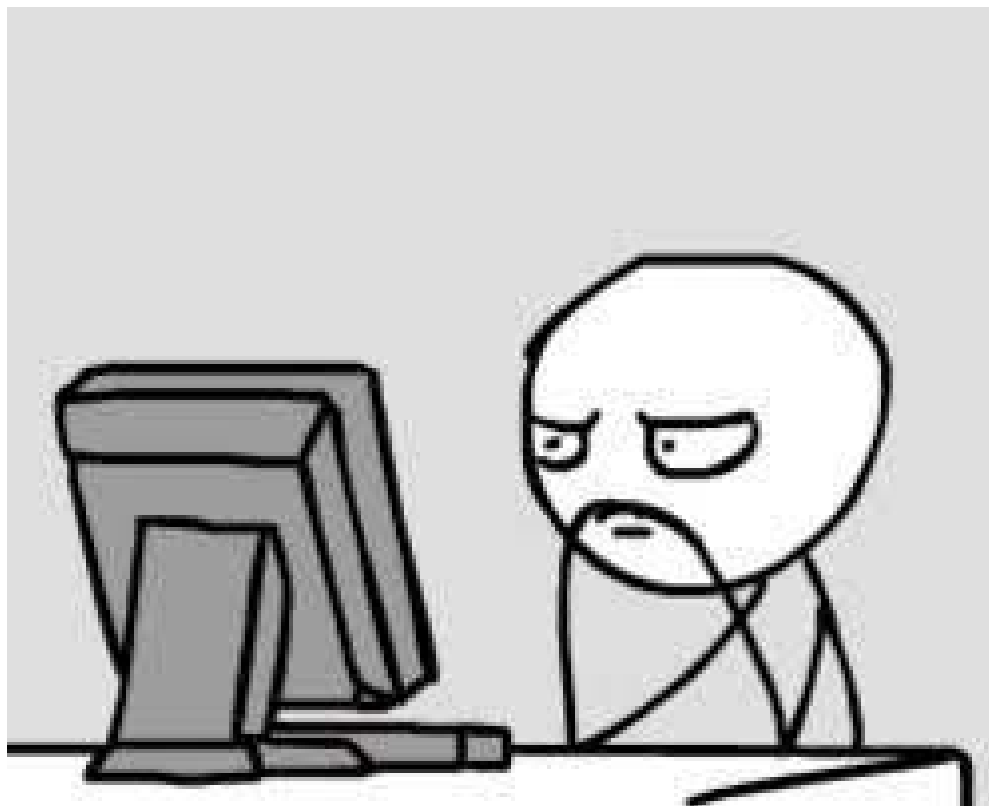
Sampling columns of X

Double decorrelation of trees and reduction of variance

Tuning parameters (bagging and random forests)

- **ntrees (# of trees / samples)**
Sensible default: 500 (bagged and RF ensembles hardly overfit)
- **nodesize & maxnodes (treesize)**
Sensible defaults:
 - nodesize: 1 for classification, 5 for regression
 - maxnodes: no limit (ensembling trees reduces variance, so trees may as well be large, with low bias)
- **mtry (# of predictor vars used for split selection)**
Sensible default: $mtry = p$ for bagging, $mtry = \sqrt{p}$ for RF
- **replace (sampling with or without replacement)**
Sensible default: Bootstrap sampling ($replace = TRUE$)
Bootstrapping increases likelihood of noise variables being selected. Subsampling ($replace = FALSE$) may perform better in terms of variable inclusion frequencies (De Bin et al., 2014).

R code example



Out-of-bag error estimation

When trees are grown on bootstrap or sub samples of the data, we do not need separate test data for error estimation:

- For each training observation i , predict Y using trees that were grown on samples not including observation i (out-of-bag)
- Then calculate as usual: $MSE = \frac{1}{n} \sum_{i=1} (y_i - \hat{y}_i)^2$

Interpretation

- ▶ The price for the better predictive accuracy of ensembles, compared to single trees, is interpretability
- ▶ Variable importances aid in interpretation of the model
- ▶ For bagging and random forests, relative importance of variable X_j is assessed as follows:
 - ▶ For every tree in the ensemble:
 - 1) Predictive accuracy for OOB observations is computed
 - 2) Predictive accuracy for OOB observations, with values of X_j randomly permuted are computed
 - ▶ Difference between accuracies under 1) and 2), averaged over all trees in the ensemble, is used to calculate importance of X_j

Boosting

- ▶ The variance of the ensemble is reduced more when trees (and predictions) are less correlated
 - ▶ Bagged ensembles decorrelate through sampling of observations
 - ▶ Random forests doubly decorrelate through sampling observations as well as variables
 - ▶ Boosting decorrelates trees through growing them sequentially: each tree is grown on the residuals of earlier trees

Pseudo response variable in boosting

In every step b of the boosting algorithm, a tree is fitted on pseudo response y_b^* , instead of the original response y :

$$y_b^* = y - \sum_{i=1}^{b-1} \nu f_i(\mathbf{x})$$

where ν is the learning rate

Tuning parameters (boosting)

Boosting does overfit, have to determine optimal parameter values by CV!

- interaction.depth (tree depth):
 - Should be relatively low (boosting works best with weak learners)
 - Rule of thumb: interaction.depth = 4
- n.trees (# of trees):
 - Rule of thumb: at least 1 / shrinkage
- shrinkage (learning rate):
 - At least 1 / n.trees
- bag.fraction (fraction of training observations randomly selected for each sample)
 - Bootstrap sampling (bag.fraction < 1) or no sampling (bag.fraction = 1)

Question

How can bagging and boosting be used for other methods than tree-based methods?

For instance, how can these be applied for linear regression and logistic regression and when is it especially beneficial to use bagging and boosting for these type of analyses?

Ensemble methods: general view

- Assume that the data are realizations of random variables:

$$(X_1, Y_1), \dots, (X_n, Y_n)$$

with p-dimensional predictor variables X and one-dimensional response variable Y

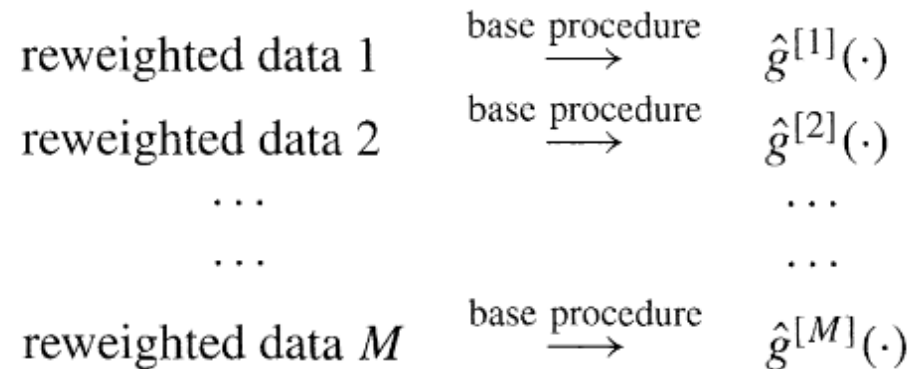
- We specify one (or more) base procedures, which produce a function estimate (that is, it *maps* input variables X to an output variable Y):

$$(X_1, Y_1), \dots, (X_n, Y_n) \xrightarrow{\text{base procedure}} \hat{g}(\cdot)$$

this base procedure can be any type of learner: tree, linear (identity) function, non-linear function, ...

Ensemble methods: general view

- Ensemble methods construct multiple function estimates or predictions from reweighted data:



- ... and use a linear combination thereof for producing the final, aggregated estimator or prediction:

$$\text{aggregation: } \hat{f}_A(\cdot) = \sum_{m=1}^M \alpha_m \hat{g}^{[m]}(\cdot).$$

Ensemble methods: general view

Ensemble methods differ in:

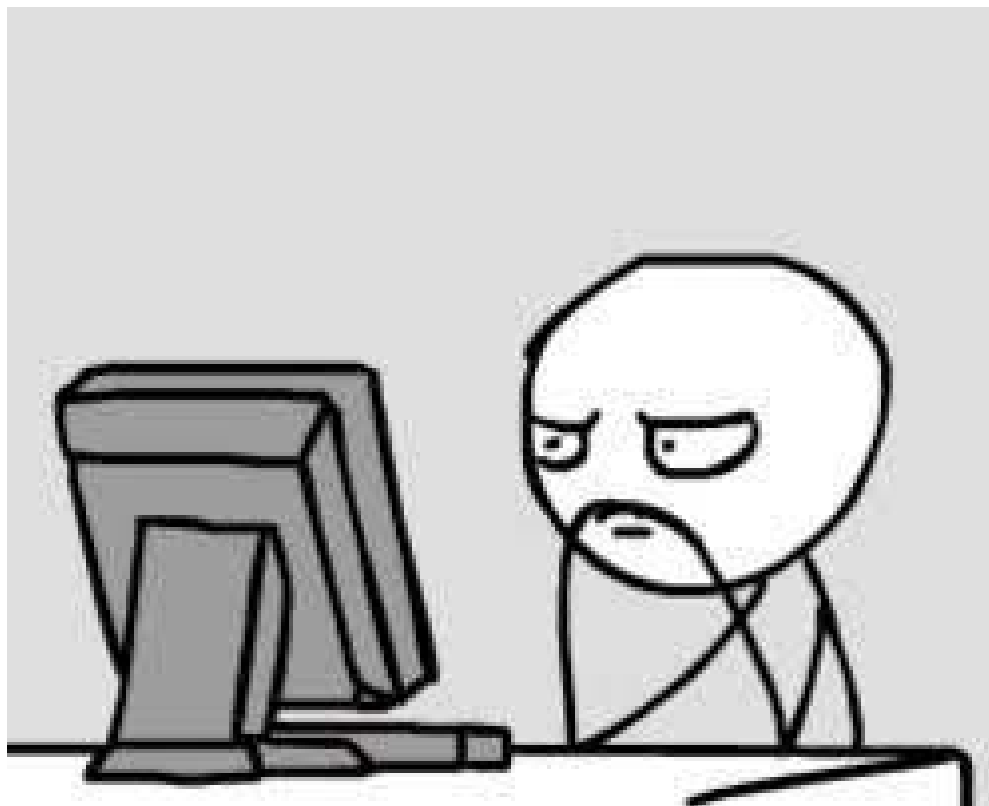
- the type of base procedures learners used
E.g., Tree, linear (identity) function, non-linear function (e.g., polynomial, hinge, interaction), support vector machine, ...
- the way the data are reweighted
E.g., bootstrap or sub sampling (e.g., bagging and random forests)
sequential reweighing, weights in iteration m depend on the results from previous iteration ($m-1$) (e.g., boosting)
...
- the way the predictions are aggregated into the ensemble:

$$\text{aggregation: } \hat{f}_A(\cdot) = \sum_{m=1}^M \alpha_m \hat{g}^{[m]}(\cdot).$$

E.g., for bagging and random forests, $\alpha_m = 1/M$;
for boosting, α_m is equal to the learning rate (tuning parameter);
coefficients α_m may be determined through penalized regression (e.g., prediction rule ensembles)

...

R code example



Trees and ensembles:

Concluding remarks

- ▶ With a single tree, we trade in some predictive accuracy for interpretability
- ▶ Depends on the context of the application if this is worth it, e.g.
 - ▶ Simple decision trees are pre-eminently suited as tools for human decision making (i.e., limited time and computational power)
 - ▶ Tree ensembles generally use all of the predictor variables: evaluation of all variables for new cases may be expensive / infeasible
- ▶ If we have lots of data and computation power available, tree ensembles provide very good predictive accuracy
- ▶ In data with little or no noise, boosting often outperforms random forests
- ▶ In noisy data, random forests often outperform boosting
- ▶ Boosting requires careful parameter tuning to perform well, random forest works well out-of-the-box
- ▶ Boosting, bagging and random forest approaches can be combined
 - ▶ e.g., `gbm()` function performs boosting with bootstrap sampling, by default

Exercises: 3 & 4 (and 5 if time permits)

References

De Bin, R., Janitza, S., Sauerbrei, W., & Boulesteix, A. L. (2015). Subsampling versus bootstrapping in resampling-based model selection for multivariable regression. *Biometrics*, 72, 272-280.

Suggested further reading:

Introduction to trees and ensembles:

Strobl, C., Malley, J., & Tutz, G. (2009). An introduction to recursive partitioning: rationale, application, and characteristics of classification and regression trees, bagging, and random forests. *Psychological Methods*, 14(4), 323.

Unbiased recursive partitioning:

Hothorn, T., Hornik, K., & Zeileis, A. (2006). Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical statistics*, 15(3), 651-674.

Boosting and ensemble learning:

Bühlmann, P., & Hothorn, T. (2007). Boosting algorithms: Regularization, prediction and model fitting. *Statistical Science* 22(4), 477-505.

Trees for multilevel data:

Fokkema, M., Smits, N., Zeileis, A., Hothorn, T. & Kelderman, H. (in press). Detecting treatment-subgroup interactions in clustered data with generalized linear mixed-effects model trees. *Behavior Research Methods*, <https://doi.org/10.3758/s13428-017-0971-x>.