Decision trees

- Aim: Separate the predictor variable space into areas R_m (subgroups) which have increasingly similar values on the response variable.

- Finding globally optimal partition: Computationally infeasible

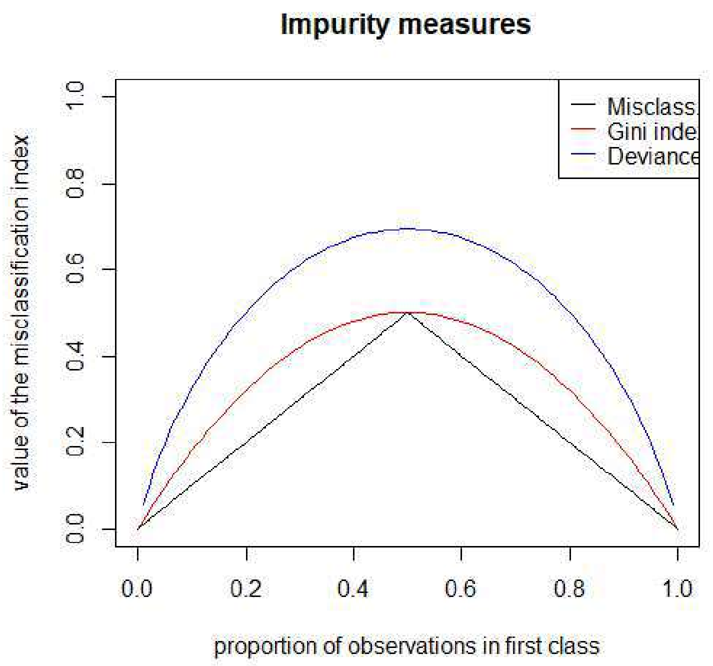
- Turn into feasible task:
  - R_m are *rectangular* and *non-overlapping*
  - Splits are recursive and involve one variable per split
  - Two-way splits only
  - Splits are found in 'greedy' fashion: split minimizing current loss is selected

- Double-edged sword: Resulting structure easy to visualize as a binary decision tree

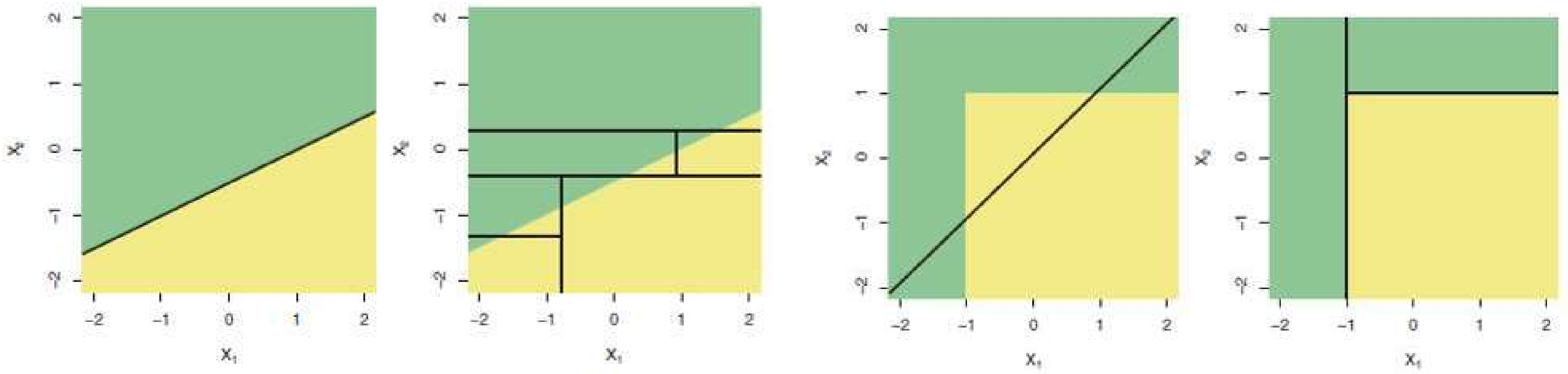
Loss functions (impurity measures)

To quantify the (dis)-similarity on the response variable, we can use several measures:

- Squared error loss: $\frac{1}{N_m}\sum_{x_i \in R_m}(y_i - \hat{y}_m)^2$
- Absolute error loss: $\frac{1}{N_m}\sum_{x_i \in R_m}|y_i - \hat{y}_m|$
- Misclassification error: $1 - \max_k \hat{p}_{mk}$
- Gini index: $\sum_{k=1}^{K}\hat{p}_{mk}(1 - \hat{p}_{mk})$
- Cross-entropy or deviance: $-\sum_{k=1}^{K}\hat{p}_{mk}\log\hat{p}_{mk}$



Decision trees: (In)stability



- The larger the tree, the lower the bias and the higher the variance (instability)
- (Note: Amount of bias not only a function of the flexibility of the method, but also a function of (shape of) true associations in data)

Decision trees: (In)stability

- Thus, again we optimize (loss + penalty):

$$\sum_{m=1}^{|T|} N_m Q_m(T) + \alpha |T|$$

- Find α through k-fold CV (note: trees (quite) different in each fold)

Variable selection bias

- CART tends to prefer splitting variables with more possible cutpoints
- Solution: Separate variable and cutpoint selection
- E.g., conditional inference trees:

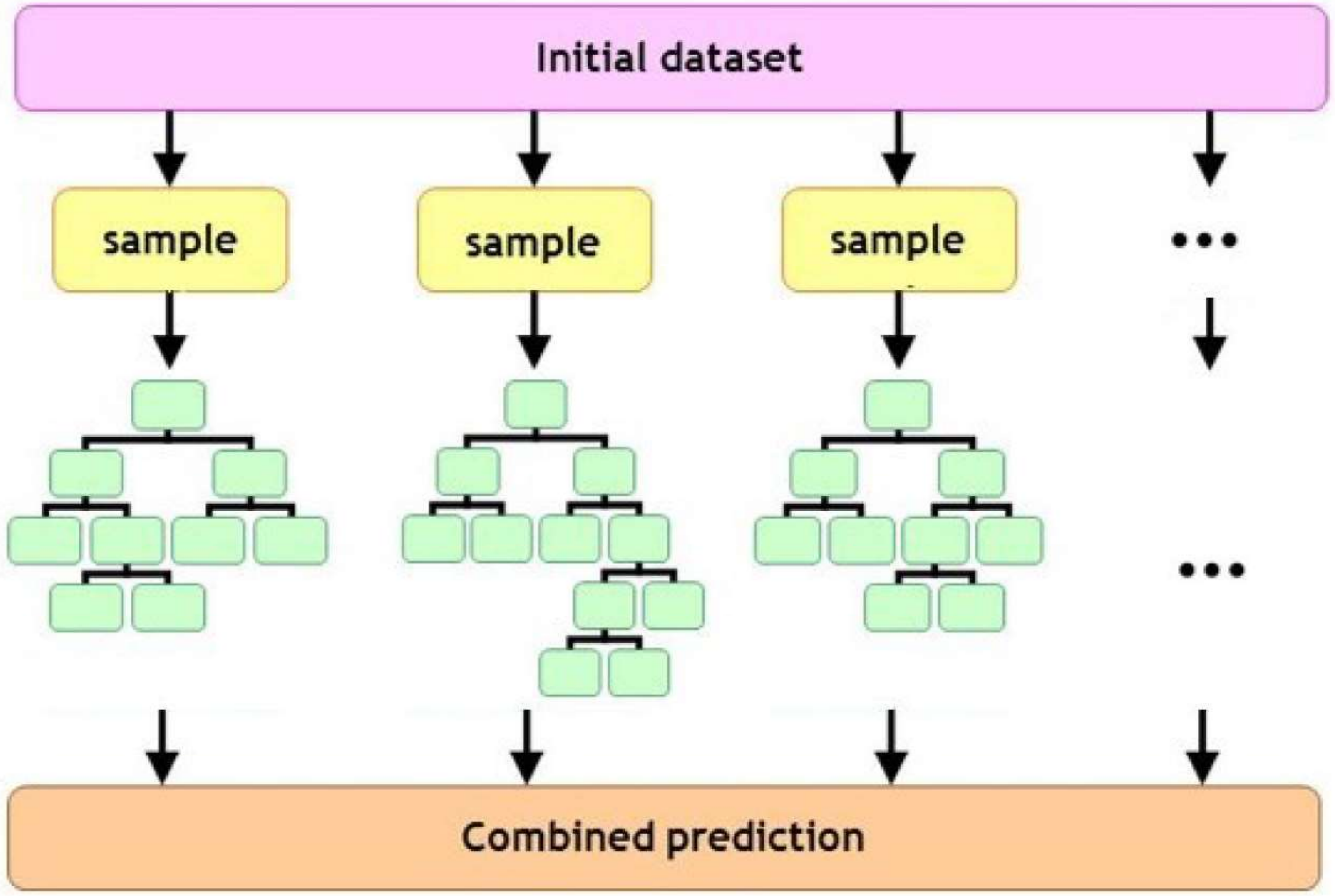
  1) Select splitting variable based on statistical association tests

  2) Select splitting value that minimizes loss (as usual)
- Uses conditional inference tests, developed by Strasser and Weber (1999)
- Tests provide natural stopping criterion for splitting

Single trees

- Good: Interpretability
- Bad: Not most accurate prediction method
- Ugly: Instability

Ensembling trees



Bagging

Draw B samples from the (initial) training dataset

- Use bootstrap (bagging) or sub sampling (subagging)
- Bootstrap yields higher inclusion frequencies for noise variables (De Bin et al., 2014)
  - Mostly disadvantageous for interpretation, less for
- Fit a tree $\hat{f}_{*b}(x)$ on each sample
- Final predictive model takes average over individual trees' predictions:

$$\hat{y} = \frac{1}{B} \sum_{b=1}^{B} \hat{f}_{*b}(x)$$

- Note: Predictions of individual trees are class labels for binary outcomes.

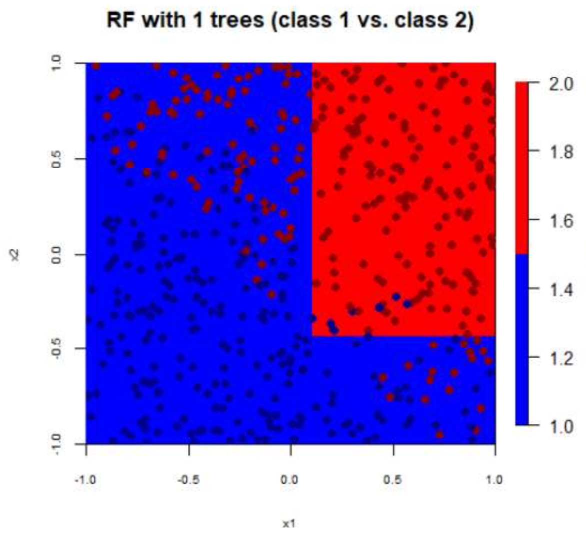
Random forest

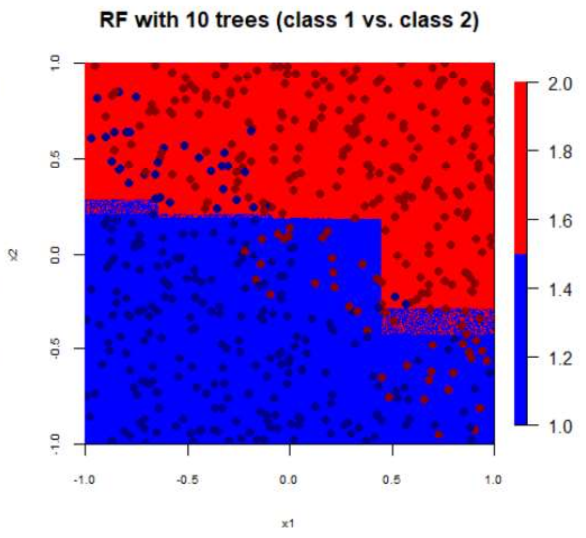
Same as bagging, in addition:

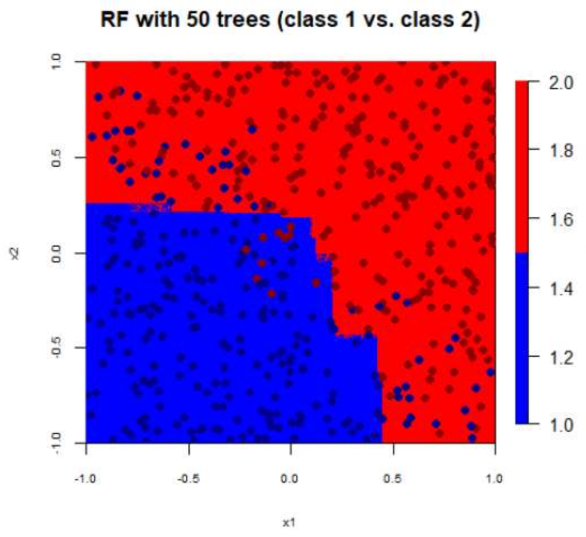
- Select random sample of *mtry* candidate predictors for every split
- Random sampling of rows as well as columns:
- Trees become more dissimilar, thus less correlated (remember: we take advantage of *instability*!)
- Allows correlated predictors to also be selected for splitting
- Final predictive model is again:

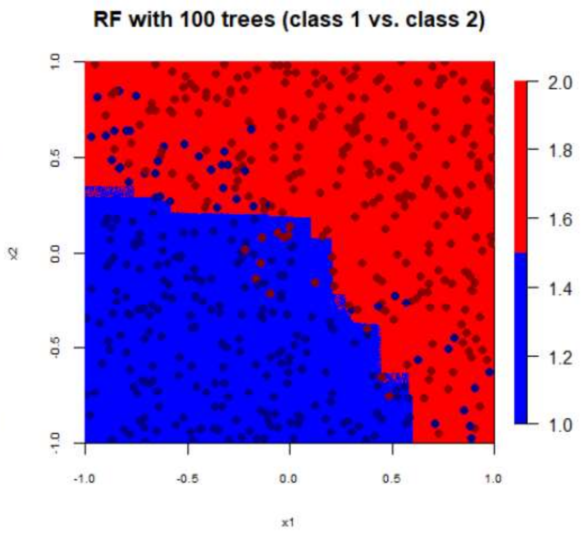
$$\hat{y} = \frac{1}{B} \sum_{b=1}^{B} \hat{f}_{*b}(x)$$

Ensembling trees performs smoothing









Out-of-bag (OOB) error

Can be computed for every baselearner (trees) fitted on samples of the training data:

- Compute OOB predictions: For every training observation i, get predicted values from each tree, fitted on samples *excluding* observation i.

- Take the average (or majority vote) to obtain the OOB prediction $\hat{y}_i^{OOB}$

- Compute MSE (or other error measures) on OOB predictions as usual:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2$$

- While OOB error is computed using training data, it provides better (less optimistic) estimate of generalization error than training error.

Interpretation: Variable importance

Importance of a predictor variable j can be computed in many (!) ways. E.g.:

- Training error: Sum over the error reductions resulting from each split involving variable j (comparable to sums of squares in GLMs)

- OOB permutation importance:

  1. Compute OOB error MSE_{OOB}
  2. For each variable j, randomly permute values of variable x_j
  3. Again generate predictions and compute MSE_{OOB}
  4. Difference between the MSE_{OOB} under 1) and 3) is the importance of variable j

Variable importances

Use with care:

- ”Importance” of a variable may sound exactly like what we want to know!
- But importances merely quantify contribution of a variable to the predictions of a given fitted model
- Thus, importance of the same variable will differ between different fitted models
- Importances are not so well defined as e.g., linear regression coefficients
- Behavior under multicollinearity, higher-order interactions may not be as expected (e.g., Strobl et al., 2007, 2008; Nicodemus et al., 2010)
- Thus, use only as a rough and approximate ordering of relevance.

- Different packages may compute importances in a different manner. Always consult help files!

Interpretation: Partial dependence functions

- The effect of a predictor variable can be computed using a partial dependence function:
$$\bar{f}(x_j) = \frac{1}{n}\sum_{i=1}^{n} \hat{f}(x_j, x_{i\setminus j})$$

- where x_j indicates the predictor variable of interest, and $x_{\setminus j}$ are all the remaining variables.

- This computes *marginal* effects (cf. GAMs, where effects are additive, and conditional effects can be computed)

- Like variable importances, partial dependence plots should be interpreted with care:
  - Possible interactions of x_j with other variables are averaged over.
  - Different packages may use different computation strategy.

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.

Nicodemus, K. K., Malley, J. D., Strobl, C., & Ziegler, A. (2010). The behaviour of random forest permutation-based variable importance measures under predictor correlation. *BMC Bioinformatics, 11*(1), 110.

Strobl, C., Boulesteix, A. L., Zeileis, A., & Hothorn, T. (2007). Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC bioinformatics, 8*(1), 25.

Strobl, C., Boulesteix, A. L., Kneib, T., Augustin, T., & Zeileis, A. (2008). Conditional variable importance for random forests. *BMC Bioinformatics, 9*(1), 307.

Boosting

Bagging and random forests:

- Fit large trees (have little or no bias)
- Average over trees to reduce variance

Boosting:

- Sequential tree fitting
- Small trees (*weak* learners; have low variance)
- Averaging over trees reduces bias, while keeping variance low

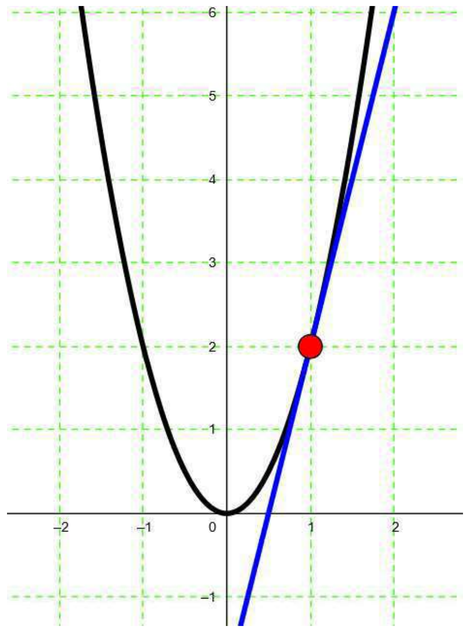
Boosting (sequential learning) can be applied to many types of learners, but particularly effective with simple, *weak* learners, like trees.

Algorithm 8.2 *Boosting for Regression Trees*

```
1. Set $\hat{f}(x) = 0$ and $r_i = y_i$ for all $i$ in the training set.
2. For $b = 1, 2, \ldots, B$, repeat:
    (a) Fit a tree $\hat{f}^b$ with $d$ splits ($d + 1$ terminal nodes) to the training
        data $(X, r)$.
    (b) Update $\hat{f}$ by adding in a shrunken version of the new tree:
            $\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x).$    (8.10)
    (c) Update the residuals,
            $r_i \leftarrow r_i - \lambda \hat{f}^b(x_i).$    (8.11)
3. Output the boosted model,
            $\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x).$    (8.12)
```

Gradient descent

- Iterative procedure for minimizing a function
- E.g., loss function $L(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$
- Take repeated steps in the opposite direction of the (approximate) gradient of the function at the current estimate:



x-axis: $\hat{y}$
y-axis: loss, e.g. $(y - \hat{y})^2$

Tree methods: Closing thoughts

- Random forests work well out of the box (e.g., default settings).
- Boosting can perform very well, and likely outperforms random forest, if parameters are carefully tuned.
- Predictive accuracy should never be sole determinant in model selection. E.g.:
- Single trees are flexible methods, easy to interpret and apply, often use lower number of variables for prediction, but lower predictive accuracy.
- Tree ensembles improve predictive accuracy, but at cost of lower interpretability and more variables required for making a prediction.
- Thus: Compare performance of simpler and more complex methods, evaluate if gain in predictive accuracy outweighs costs.

Tuning parameters: Boosting

- `shrinkage` (λ, learning rate): Small, non-zero values; the default of 0.1 is quite high, values of .01 or .001 often better.

- `n.trees`: Number of random samples (trees) to generate. Generally requires values $> (1/\lambda)$.

- `distribution`: Specifies which loss function should be minimized, and thereby how the gradient is computed. For continuous responses, specify `"gaussian"` (or e.g., `"laplace"`, to minimize absolute error loss). For binary responses, use `"bernoulli"`. See `?gbm` for more options.

- `interaction.depth` (tree depth; $\neq$ number of splits or nodes). Specifies the level of variable interactions allowed; default is 1, yielding trees with only a single split, i.e., an additive model.

- `bag.fraction`: Fraction of training observations randomly selected to fit each tree. A value of 1 will result in no subsampling, and all training observations used for inducing each tree. Default is 0.5, yielding a subsample comprising 50% of the training observations.

Tuning parameters: Bagging and random forests

- `ntree`: Number of samples to draw / trees to generate. Default of 500 often suffices, rarely hurts to use more (e.g., 1,000).
- `mtry`: Number of predictors to be considered for each split; p for bagging; random forests use $\sqrt{p}$ for classification, $p/3$ for regression. Can use CV to obtain optimal value.
- `replace` and `sampsize`: Sampling strategy and fraction. Bootstrap sampling is sensible default, but: may increase inclusion frequency for noise variables. Subsampling with sampling fraction of .632 may be preferred.
- `maxnodes`: Maximum number of nodes in every tree. By (sensible) default limited by `nodesize` (1 for classification and 5 for regression). But: large trees can lead to unstable results when there are many multicollinear predictors at best weakly related to the response (Segal, 2003). Thus, smaller trees (with optimal tree size e.g., determined through CV) may perform better.