Statistical Learning

Tommaso Proietti

DEF Tor Vergata

Additive Models and Trees

Additive Models

- Let us consider the regression model with multiple inputs:

$$Y = f(X_1, X_2, \ldots, X_p) + \epsilon,$$

  where $f(X_1, X_2, \ldots, X_p) = \mathrm{E}(Y|X_1, X_2, \ldots, X_p)$.
- We specify

$$\mathrm{E}(Y|X_1, X_2, \ldots, X_p) = \alpha + f_1(X_1) + f_2(X_2) + \cdots + f_p(X_p).$$

- The functions $f_j(X_j)$ are smooth nonparametric functions, e.g. cubic smoothing splines, of a single input.
- This approach stands somewhat between linear (and additive) regression and the non-linear non-additive regression approach, which is too general and prone to the curse of dimensionality.

- A training sample is available $\{(y_i, \boldsymbol{x}_i), i = 1, \ldots, N\}$, $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})'$.
- Estimation of the intercept α and the functions f_j is carried out by the following backfitting algorithm:
  0. Initialisation: set $\hat{\alpha} = \frac{1}{N}\sum_i y_i$ and $\hat{f}_j \equiv 0$
  1. Compute the partial residuals from the fit excluding the j-th function:

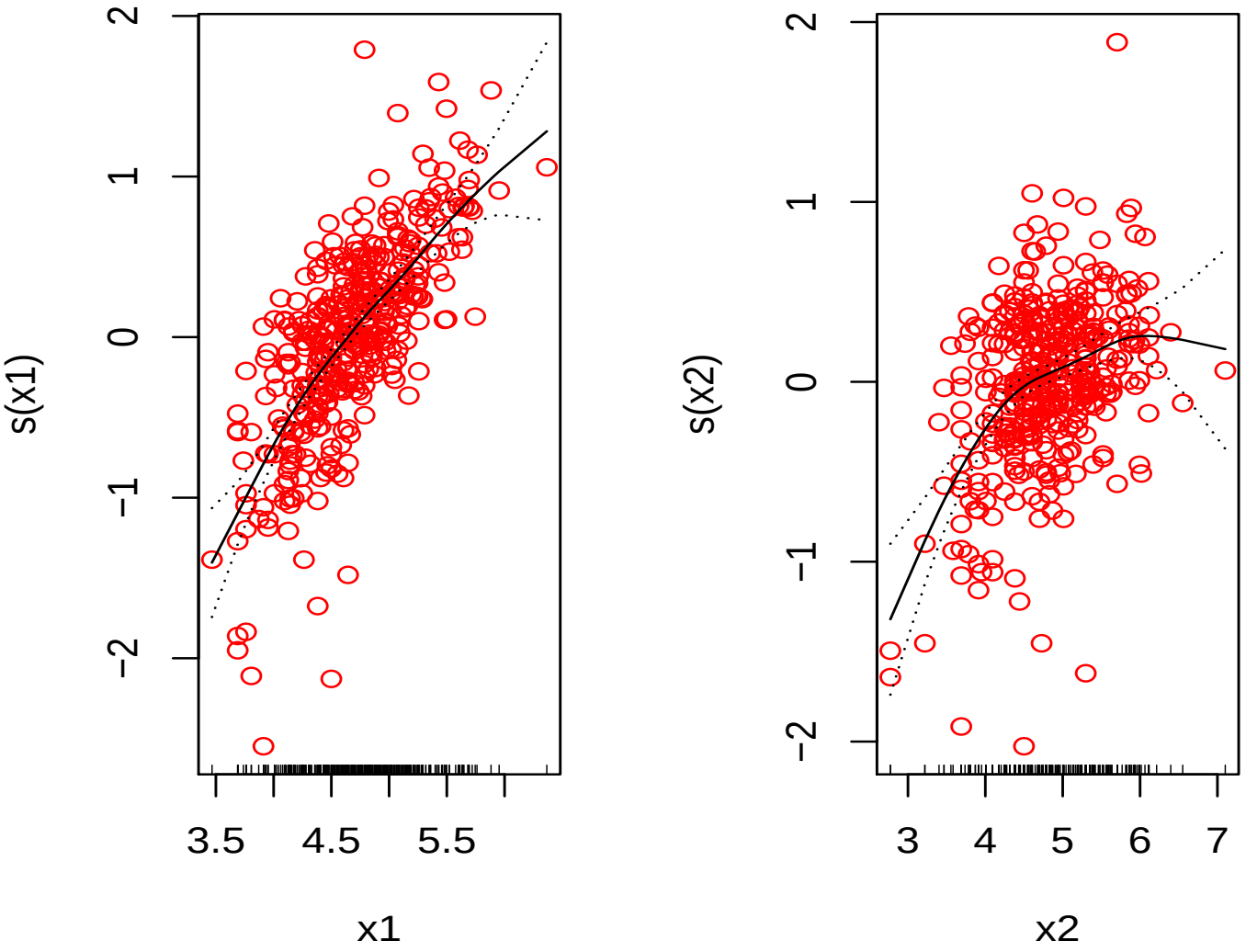
$$e_{i\setminus j} = y_i - \hat{\alpha} - \sum_{k \neq j} \hat{f}_k(x_{ik})$$

  2. Apply a smoother $\mathcal{S}_j$ to $e_{i\setminus j}$ to obtain $\hat{f}_j$
  3. Reset $\hat{f}_j$ equal to $\hat{f}_j - \sum_{i=1}^N \hat{f}_j(x_{ij})/N$

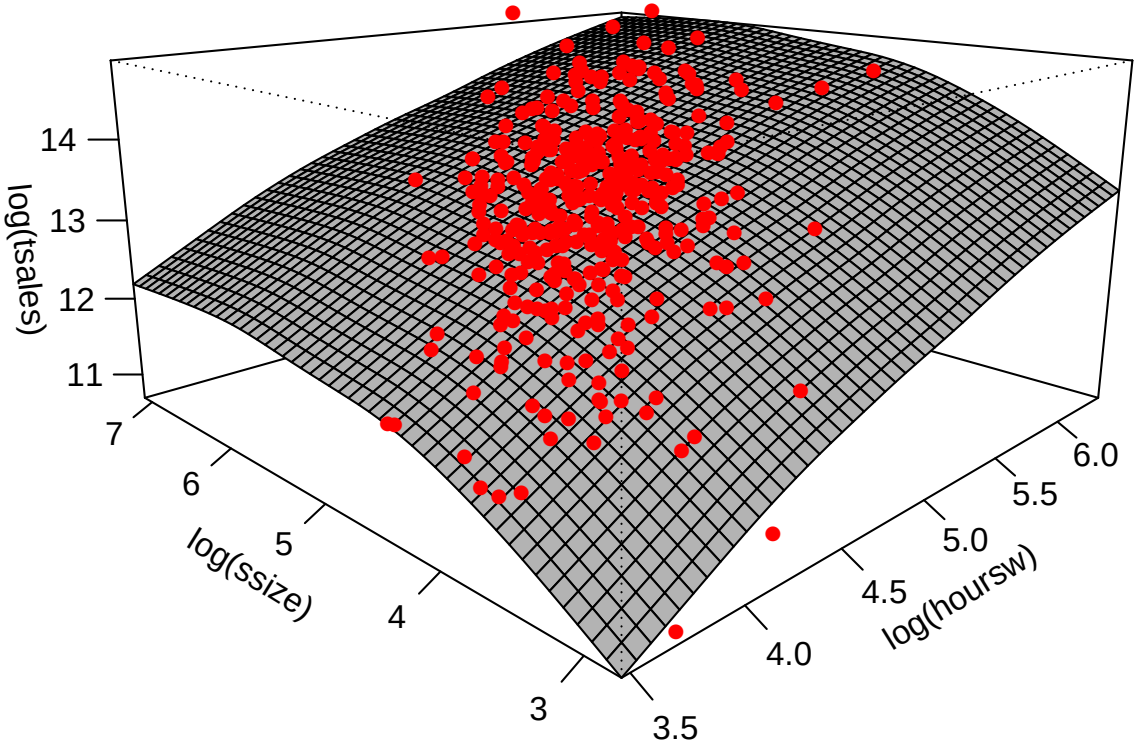
  The last step ensures that $\sum_{i=1}^N \hat{f}_j(x_{ij}) = 0, \forall j$.
- The steps 1-3 are iterated until convergence.

- The smoother $\mathcal{S}_j$ can be a cubic smoothing spline with smoothness parameter λ_j or a local polynomial smoother with bandwidth h_j.
- The selection of the parameters governing the complexity of the fit is done by minimizing the test error (e.g. evaluated by cross-validation).

Additive model using smoothing splines: `gam(y ~ s(x1)+s(x2), data = clothing)`



Additive model using splines: `gam(y ~ s(x1)+s(x2))`



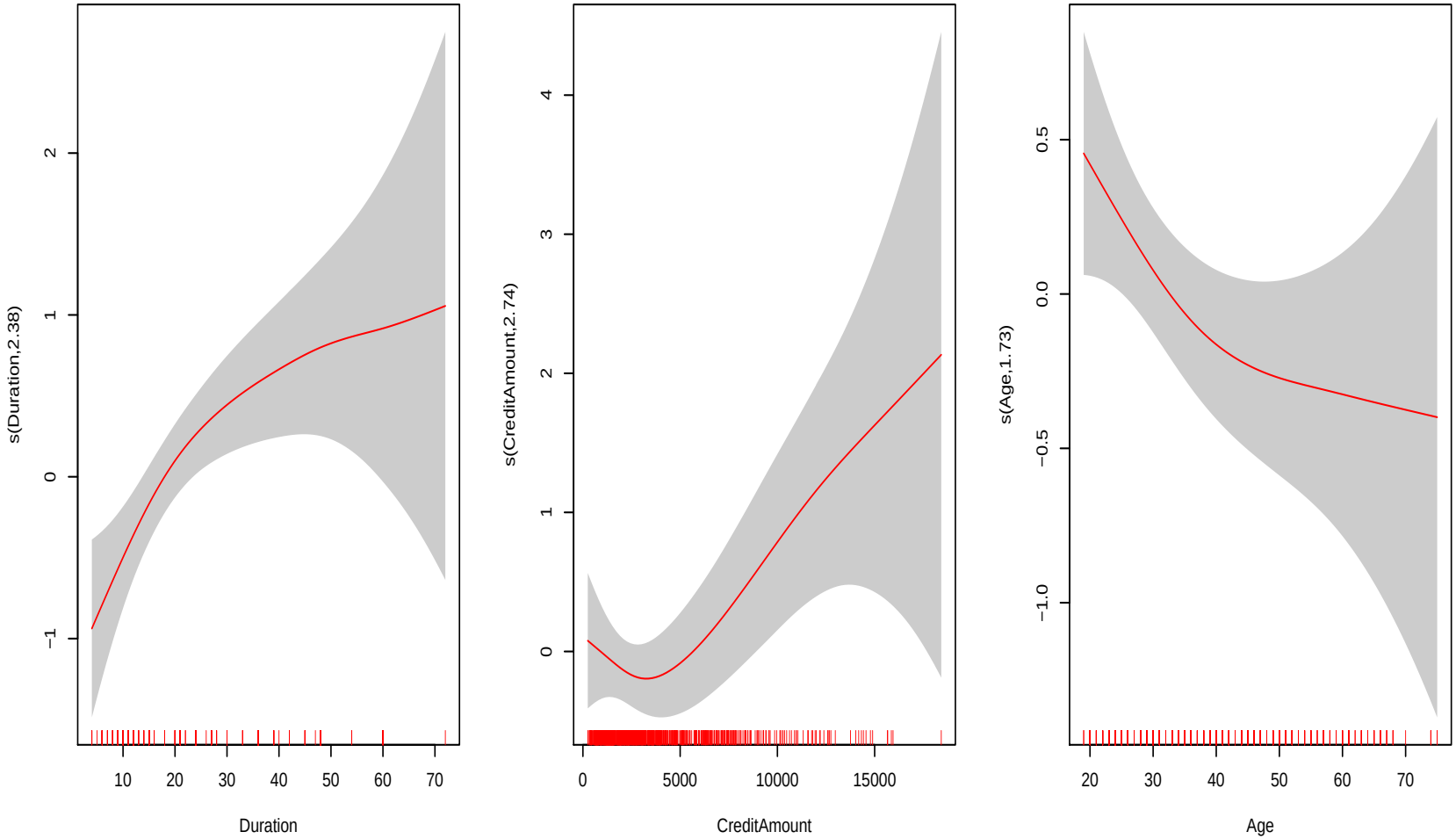
Additive Logistic Regression

The generalized additive logistic regression is additive in the log-odds:

$$\ln \frac{P(G=1|X)}{P(G=0|X)} = \alpha + f_1(X_1) + f_2(X_2) + \cdots + f_p(X_p).$$

where $f_j(X_j)$ is a smooth nonparametric function.
Estimation is carried out by the backfitting algorithm.

Additive logistic regression. `gam(Group ~ s(Duration)+s(CreditAmount) + s(Age)+factor(...), family=binomial(link="logit")`



Tree-based Methods

- Tree-based methods partition the input space into rectangles, using rules to identify regions characterized by a homogeneous response to the inputs.
- The idea is to sub-divide, or partition, the space into smaller rectangular regions and to fit a simple model, such as a constant, to the observations falling in the region.
- Basic model selection problems, like variable selection, monotonic transformations and interactions are handled by deciding which variables to use in the partitioning and how.

Regression Trees

Suppose we have p input variables $X_1, \ldots, X_p$ and a **quantitative** response variable Y.

Denote the training sample by $\{(y_i, \boldsymbol{x}_i), i = 1, \ldots, N\}$.

Let $\{R_m, m = 1, \ldots, M\}$ denote a set of disjoint rectangles that partition the input space $(X_1, \ldots, X_p)$.

A regression tree is an additive model of the form

$$f(\boldsymbol{x}) = \sum_{m=1}^{M} c_m I(\boldsymbol{x} \in R_m).$$

Conditional on the splits, the least squares estimate of c_m is
$\hat{c}_m = \text{Average}\,[y_i | \boldsymbol{x}_i \in R_m]$.
The main issue is how to determine the rectangular regions $\{R_m, m = 1, \ldots, M\}$.

A greedy top-down recursive partitioning algorithm is used. The model is fitted sequentially by performing a binary split involving a single input and can be represented as a tree.

- For any splitting variable j and a split point s we define the rectangles

$$R_1 = \{\boldsymbol{x} : x_j \leq s\}, R_2 = \{\boldsymbol{x} : x_j > s\}.$$

  We select the variable X_j and the split point s which minimise the residual sum of squares

$$\sum I(\boldsymbol{x}_i \in R_1)(y_i - c_1)^2 + \sum I(\boldsymbol{x}_i \in R_2)(y_i - c_2)^2.$$

- Repeat the splitting process in each rectangle. Each node is split in two groups.

The subsets created by the splits are called nodes. The subsets which are not split are called terminal nodes. Terminal nodes are also known as leaves of the tree. To each leave there correspond a partition.

We aim at determining the optimal size of the tree.

Suppose there are T terminal nodes and denote by N_m the number of observations belonging to each leaf, and by

$$D_m(T) = \sum I(\boldsymbol{x}_i \in R_m)(y_i - \hat{c}_m)^2$$

the deviance of the residuals associated to the leaf.

$Q_m(T) = D_m(T)/N_m$ is often referred to as squared-error node impurity measure.

$D(T) = \sum_{m=1}^{T} D_m(T)$ is the deviance for a tree with T leaves.

Usually a large tree is grown, stopping the splitting process when a threshold value for $D_m(T)$ is reached, or N_m becomes very small. Let T_0 be the size of this tree.

For selecting a tree with size $T \leq T_0$, we minimize the cost complexity, defined as

$$D(T) + \alpha T,$$

where the complexity parameter, $\alpha \geq 0$, regulates the trade-off between goodness of fit (as measured by the deviance) and the tree size.

For $\alpha = 0$ the solution is the full tree T_0. On the other hand, if $\alpha \to \infty$ the fit tends to the global mean $\bar{y}$ (no partition).

Estimation of α is done by (5 or 10-fold) crossvalidation. Given $\hat{\alpha}$, the corresponding T_α is obtained by weakest link pruning, i.e. we prune the leaves that produce the smallest increase in the deviance, when collapsed.

Building a regression tree

1. Grow a large tree on the training data by recursive binary splitting.
2. For each value of α (on a grid), obtain the best subtree, i.e., the one minimising the cost complexity, $C(\alpha) = D(T) + \alpha T$ (e.g., for $\alpha = 0$ this is the full tree, for $\alpha \to \infty$ this is the tree without leaves $T = 0$ (constant average fit)).
3. Select the optimal value of α by K-fold cross-validation. Subdivide the training sample into K folds. For $k = 1, \ldots, K$:
    1. Repeat Steps 1 and 2 for all folds except the k-th fold.
    2. Evaluate the cross-validation score on the k-th fold as a function of α, $CV_k(\alpha)$.
    3. Select the value of α for which $\sum_k CV_k(\alpha)$ is a minimum.
4. Select the subtree in step 2 corresponding to optimal value of α.

Figure: Regression tree, clothing dataset.

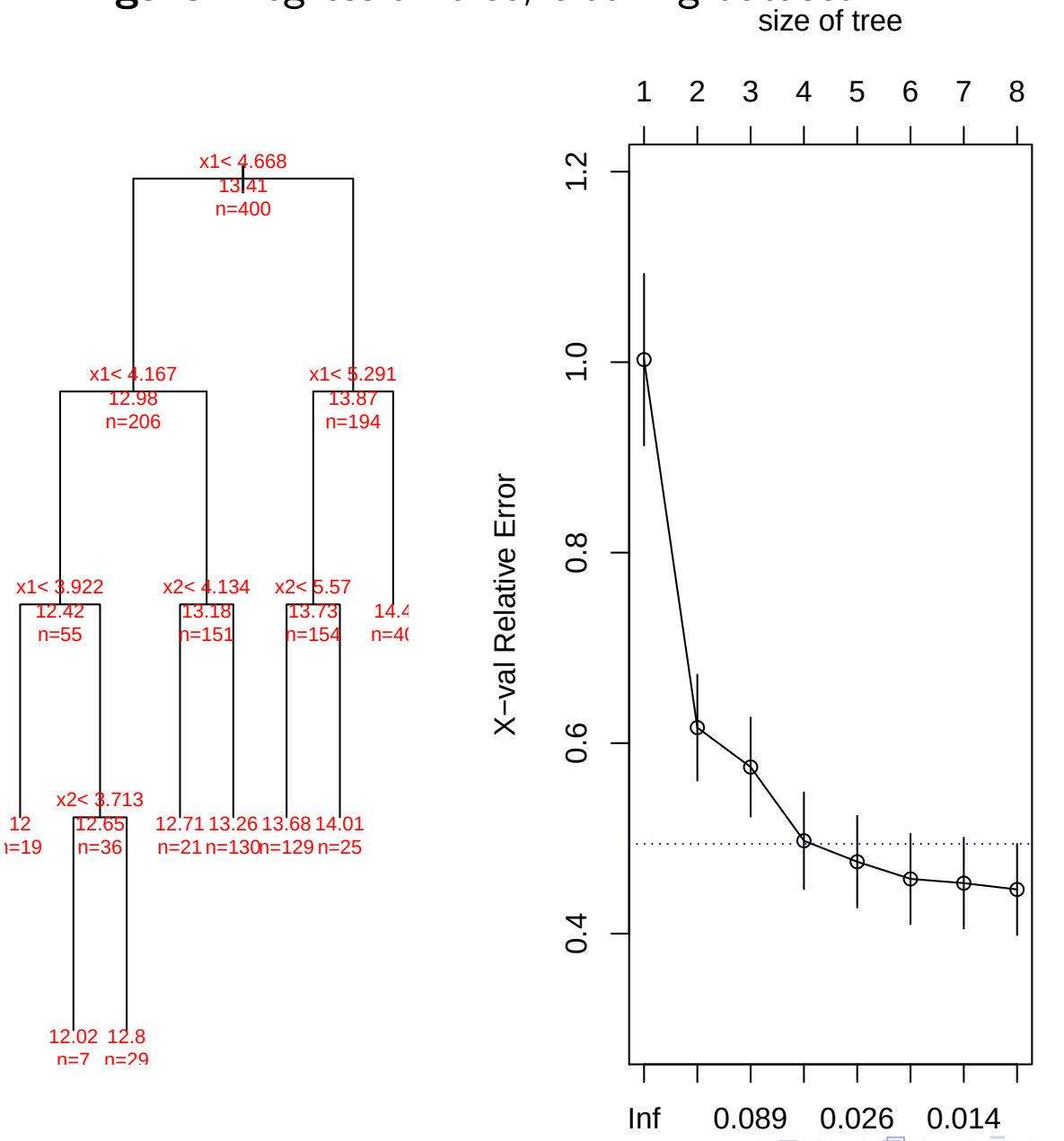
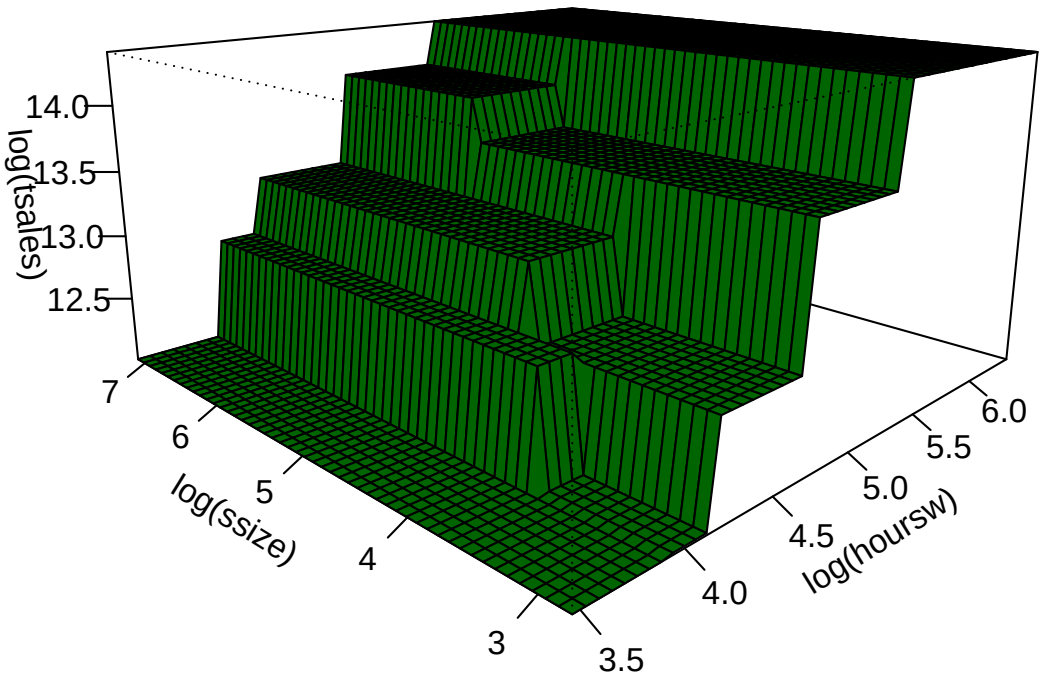


Figure: Regression tree, clothing dataset.



Classification Trees

- Let Y denote a nominal variable and let Y_k denote the dummy variables for the k-th response category.
- Given a set $\{R_m, m = 1, \ldots, M\}$ of disjoint rectangles that partition the input space, we compute the mean of $Y_k, k = 1, \ldots, K$, over the N_m units belonging to the partition; this is an estimate of $P(G = k|\boldsymbol{x} \in R_m) = p_{mk}$.
- The units falling in this region will be classified according to the usual majority rule.
- Let κ be the modal class and let $\hat{p}_{m\kappa}$ be the associated fraction of cases.
- The tree is grown using different measures of the value of a split.

Let $\hat{p}_{mk}$ denote the fraction of the observations in the region R_m belonging to class k.

Three measures of node impurity are popular:

- The missclassification error: $1 - \hat{p}_{m\kappa}$, i.e. the fraction of units missclassified in node m.
- The Gini index: $1 - \sum_{k=1}^{K} \hat{p}_{mk}^2$.
- Cross-entropy or deviance: $-\sum_{k=1}^{K} \hat{p}_{mk} \ln \hat{p}_{mk}$

Both Gini and cross-entropy are measures of the *heterogeneity* of the probability distribution $\{\hat{p}_{mk}, k = 1, \ldots, K\}$.

We look for the split that yields minimum heterogeneity.

Maximum heterogeneity corresponds to the case when $\hat{p}_{mk} = 1/K$ (the posterior probability are uniformly distributed - no majority vote), and the Gini index is equal to $1 - 1/K$ whereas cross-entropy equals $\ln K$.

On the contrary, heterogeneity is a minimum when $\hat{p}_{m\kappa} = 1$ and $\hat{p}_{mk} = 0, \forall k \neq \kappa$, in which case both indices are equal to 0.

Figure: Classification tree, German Credit dataset.

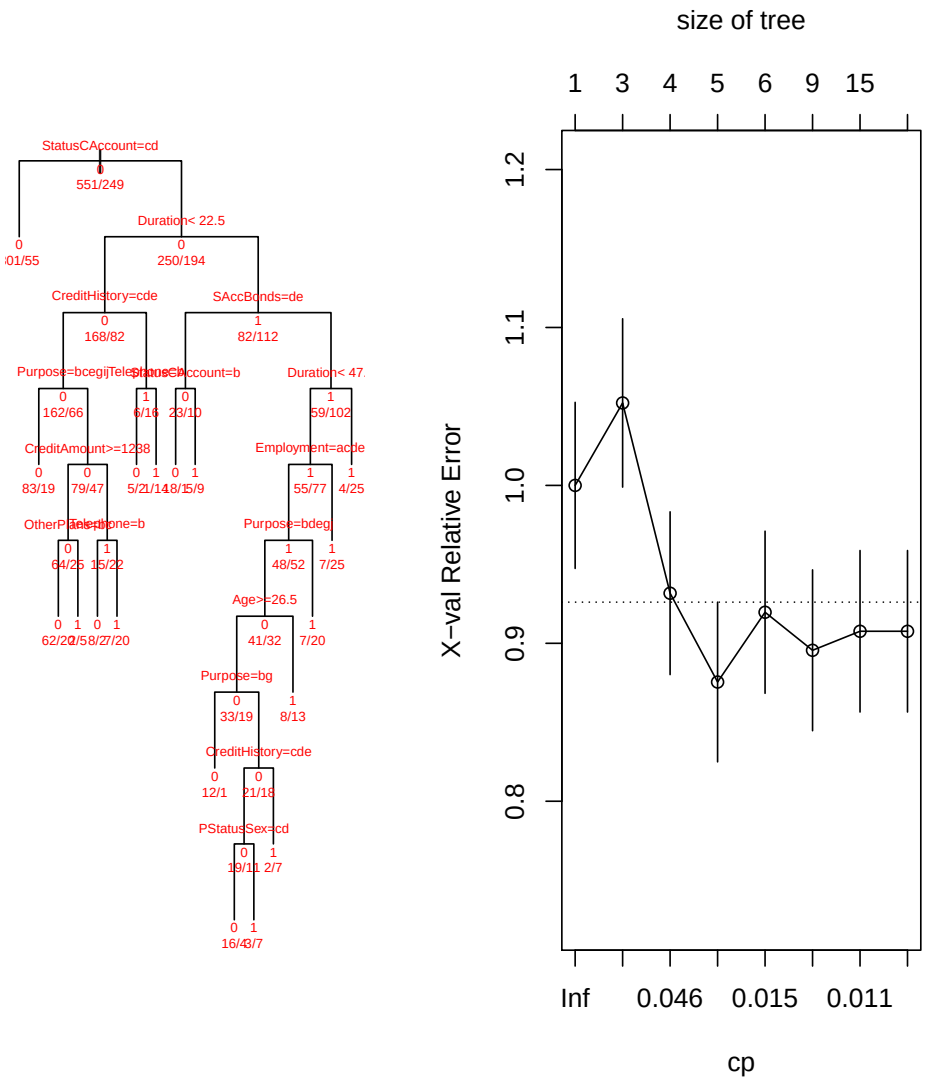
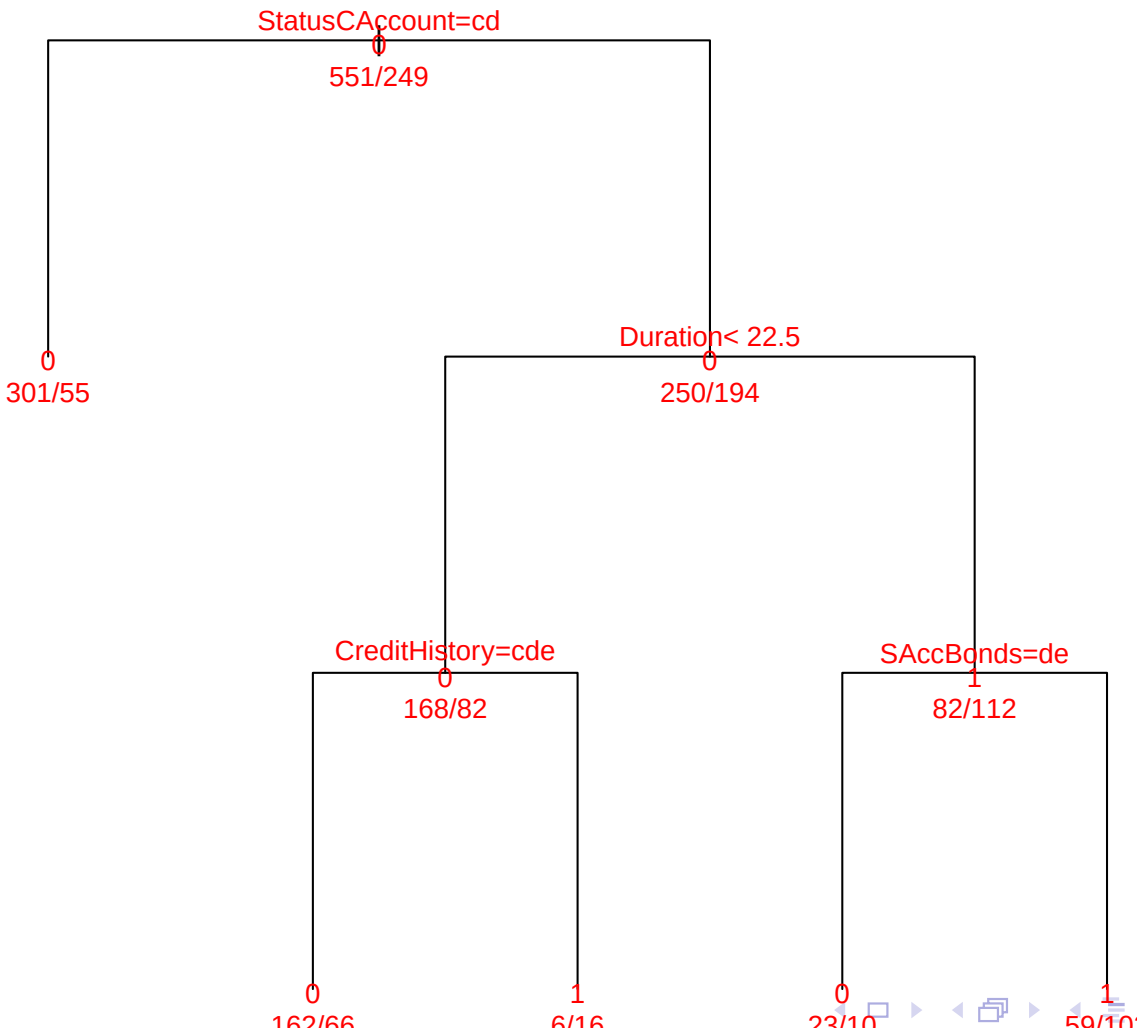


Figure: Pruned tree, Credit dataset.



Bagging (Bootstrap Aggregation)

- One serious limitation of trees is their volatility, which is related to the hierarchical structure of the splitting process. Small changes in the training sample produce a different sequence of splits.
- *Bagging*, which averages predictions over trees drawn from the same population, provides a solution, reducing the variance of the predictions.
- A *bootstrap sample* is a sample of size N drawn with replacement from the training data $(y_i, \boldsymbol{x}_i)$.
- Suppose B such samples are drawn independently. They can be used to assess the uncertainty of a method or model (parameter uncertainty, prediction uncertainty) by looking at the variability of the results.
- In our case, however, interest lies in the prediction $\hat{f}(\boldsymbol{x})$ for a unit with input feature $\boldsymbol{x}$.

For each boostrap sample we estimate the model or apply the model and compute $\hat{f}_b(\boldsymbol{x}), b = 1, \ldots, B$.

The bagging estimate of $f(\boldsymbol{x})$ is

$$\hat{f}_{bag}(\boldsymbol{x}) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}_b(\boldsymbol{x})$$

For linear methods (such as linear regression) bagging tends to the linear fit $\hat{f}(\boldsymbol{x})$ as B increases. Hence, it is of little use in those frameworks.

For regression trees, the bagged estimate is the average prediction from B trees.

For classification trees, we use bagging to estimate the probability that a unit with feature $\boldsymbol{x}$ belongs to group k, $\hat{p}_k(\boldsymbol{x})$, as the proportion of the B trees predicting class k.

Alternatively, we can average across the values $\hat{p}_{b,k}(\boldsymbol{x})$ computed on the bootstrap samples:

$$\hat{p}_{bag,k}(\boldsymbol{x}) = \frac{1}{B}\sum_{b=1}^{B}\hat{p}_{b,k}(\boldsymbol{x}).$$

General note: about one third of the training sample is out of the bag and can be used for validation (estimation of the test error).

Trees are grown deep, without pruning (low bias, high variance). The variance is reduced by averaging the trees.

Problem is that the trees are very similar. Little diversification and predictors are going to be highly correlated.

Random forests

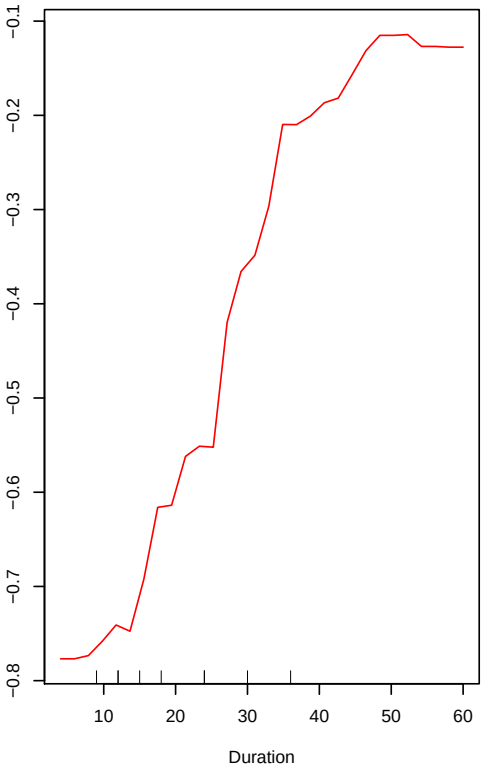
- In some situations both N (number of observations) and p (number of inputs) are very large.
- Several decision trees are grown by selecting a random subsample of both the units (with replacement) and of the inputs ($m \approx \sqrt{p}$ predictors).
- The trees are 'decorrelated', as they are built on a randomly chosen subsample of input variables.
- Each decision tree is built to its maximum size, (no pruning).
- The trees are combined into a single classifier by averaging the individual classifiers.

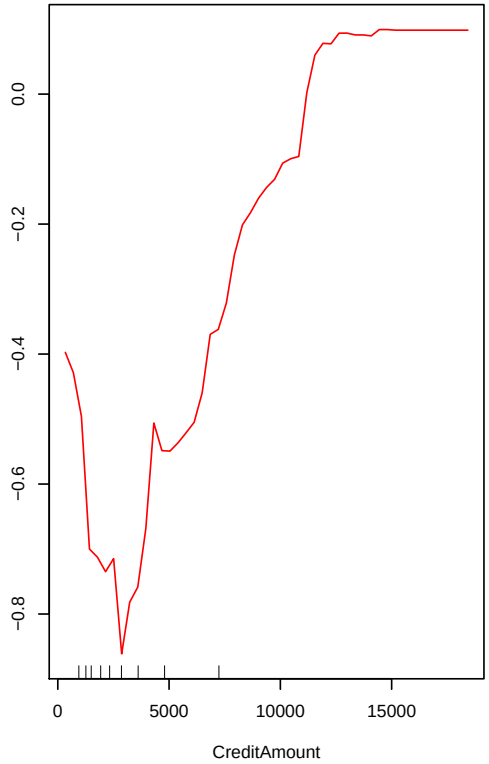
Variable importance

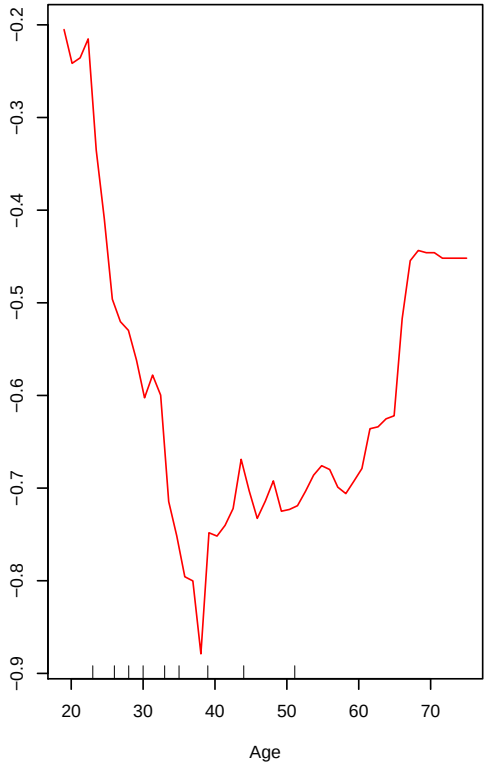
There are two essential ways of measuring the relative importance of each input variable in predicting the response.

- The *split-based variable importance* measure is based on the number of times a variable is selected for splitting, weighted by the squared improvement in the impurity measure as a result of each split, and averaged over all trees. This is subsequently scaled so that the sum adds to 100, with higher numbers indicating stronger influence on the response.
- *Partial Dependence plots* are obtained by integrating out the effects of other predictors to show the partial dependence of the fit on the predictors of interest.

Random forests. Partial dependence plots (logits) for `Duration`, `CreditAmount` and `Age`.







The receiver operating characteristic (ROC) curve

Consider the confusion matrix:

	$\hat{G}(X)$ (prediction outcome)	
G (actual value)	0	1
0	True negative (TN)	False positive (FP)
1	False negative (FN)	True positive (TP)

The true positive rate (TPR) is defined as

$$P(\hat{G}(X) = 1|G = 1) = TPR = \frac{TP}{TP + FN}$$

this is also referred to as the sensitivity rate.
The false positive rate (FPR) is defined as

$$P(\hat{G}(X) = 1|G = 0) = FPR = \frac{FP}{TN + FP}$$

The specificity rate is $P(\hat{G}(X) = 0|G = 0) = \frac{TN}{TN+FP}$;

- The **ROC curve** displays the true positive rate (TPR, aka sensitivity) on the vertical axis and false positive rate (FPR, 1-specificity) on the horizontal axis.
- The point (0,1) is the perfect classification point. The 45 degrees line is the line of no discrimination (random guess). Below the line the classifier performs worse than a pure random guess.
- ROC is used to illustrate the trade-off between TPR and FPR.
- In logistic regression $\hat{p}_i, y_i$ are sorted according to the values of $\hat{p}_i$ from the largest to the smallest. For each threshold p from 1 to 0 we compute the TPR and FPR when the classifier allocates to Group 1 if $\hat{p}_i > p$.
- The **area under the curve** (AUC) is often used as a measure of accuracy. AUC is 0.5 for a random classifier. Its maximum is 1.