DataFest - ML Intro and Trees

Bernd Bischl

Computational Statistics, LMU

My interests and research areas

Machine learning

- General non-parametric methods
- Hyper-parameter tuning and model selection

Model-based optimization, selection and tuning

- Generalzing MBO strategies to more complex tuning problems
- Parallel MBO

In short: ML + optimization in COMBINATION!

Practical stuff

- Benchmarking repositories
- Open science, sharing and reproducibility
- Efficient development of statistical software
- Parallelization

Section 1

Introduction

What is (supervised) machine learning?

- Learning structure in data
- The art of predicting stuff
- Model optimization
- Understanding of grey-box models

Introduction to Machine Learning II

- Data analytical problems increased in the last decade regarding size and complexity
  - data mining: storage, organization and analysis of big data sets
  - bio informatics: solving statistical and computational problems in medicine an biology
- Role of statistics: recognition of important patterns and trends, attempt to understand what "data reveals", creation of predictions
- New York Times (August 2009): *"'I keep saying that the sexy job in the next 10 years will be statisticians,"' said Hal Varian, chief economist at Google. "'And I'm not kidding."'*

Supervised Learning

- Try to learn the relationship between "input" x and "output" y.
- For learning, there is training data with labels available.
- Considered mathematically, both cases are problems of function-approximation: search for an f, such that

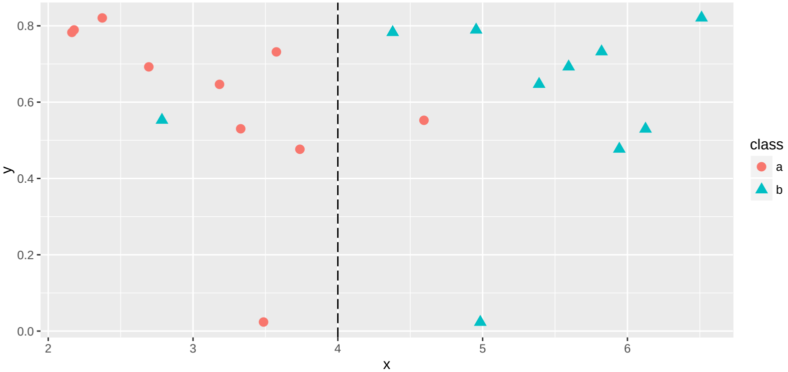
$$y \approx f(x).$$

Introduction to Machine Learning IV

Examples

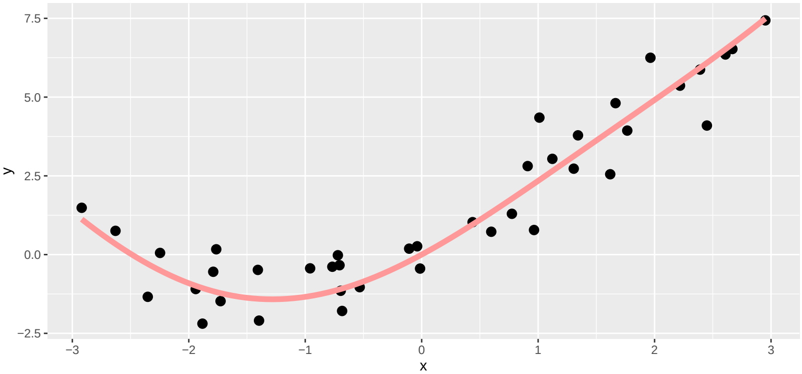
- Handwritten digit recognition
- Lung cancer prediction
- Email spam recognition
- Recommender system (movies, books, etc.)
- Word recognition from spoken language
- ...

Supervised Classification tasks



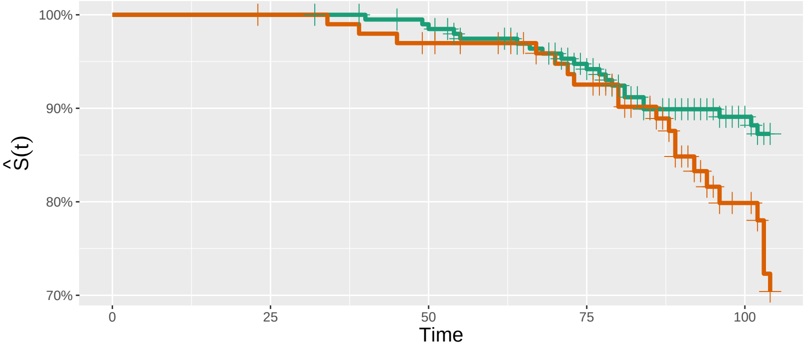
Goal: Predict a class (or membership probabilities)

Supervised Regression tasks



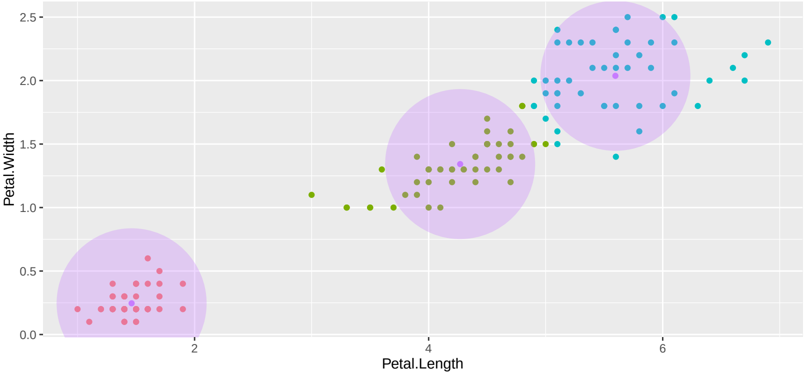
Goal: Predict a continuous output

Supervised Survival tasks



Goal: Predict a survival function $\hat{S}(t)$, i.e. the probability to survive to time point t

Unsupervised Cluster tasks



Goal: Group data into similar clusters (or estimate fuzzy membership probabilities)

Section 2

Classification and Regression Trees (CART)

Trees - Introduction

- Regression and classification trees exist (and others)
- Trees divide the feature space into rectangles and fit simple models (e.g: constant) in each:

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

where M rectangles R_m are used. c_m is either the average output of the observations in R_m (regression) or the class distribution / most frequent label in R_m (classification).

Components of the algorithms

- Greedy: Pick the best feature and its best splitpoint in each iteration
- Binary splits vs. multi-way splits
- Criteria for the selection of a variable and its splitpoint(s)
- Stopping-Criteria
- Handling of missing values
- Pruning

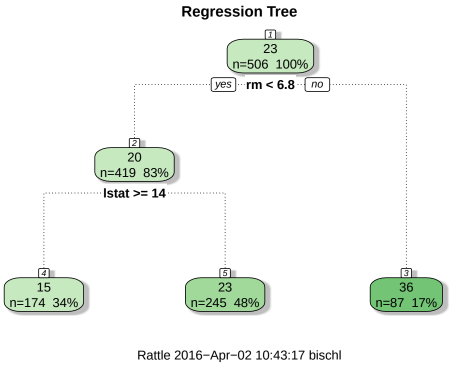
Tree Building Example I

We use two data sets for our examples:

- Regression: The `BostonHousing` data set has 506 observations (census tracts of Boston from the 1970 census) and 14 variables, `medv` (median value of the owner-occupied homes) being the target variable.
- Classification: The `iris` data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris (setosa, versicolor, and virginica).

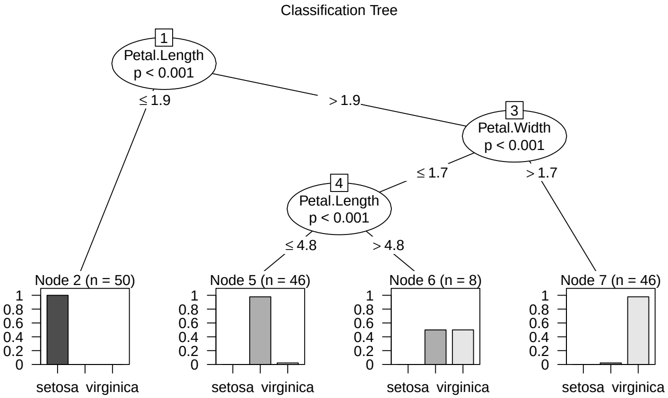
Tree Building Example II

```
library(rattle); library(rpart)
data(BostonHousing, package = "mlbench")
m = rpart(medv ~ ., data = BostonHousing, minsplit = 250)
fancyRpartPlot(m, main = "Regression Tree")
```

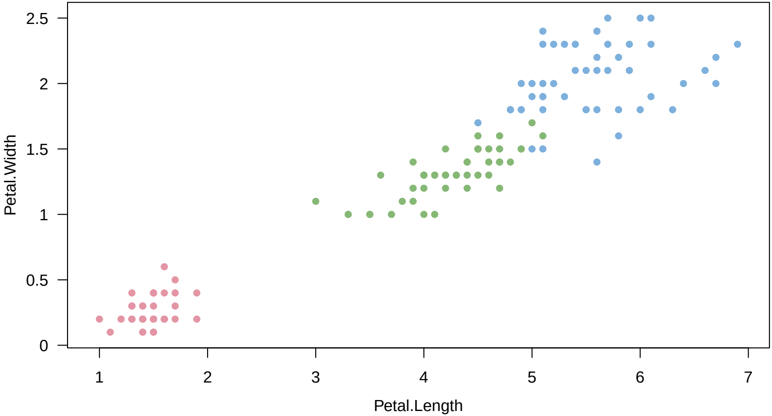


Tree Building Example III

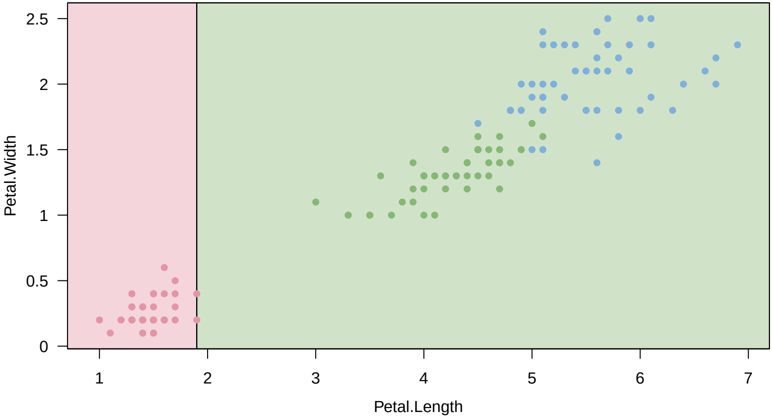
```r
data(BostonHousing, package = "mlbench")
m = ctree(Species ~ ., data = iris)
plot(m, main = "Classification Tree")
```



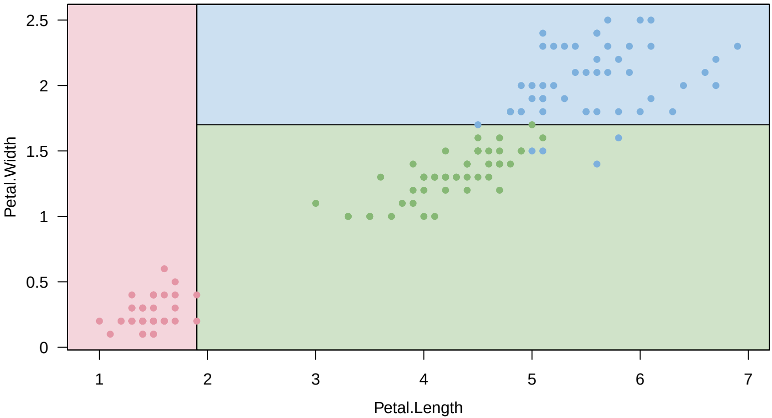
Tree Building Example IV



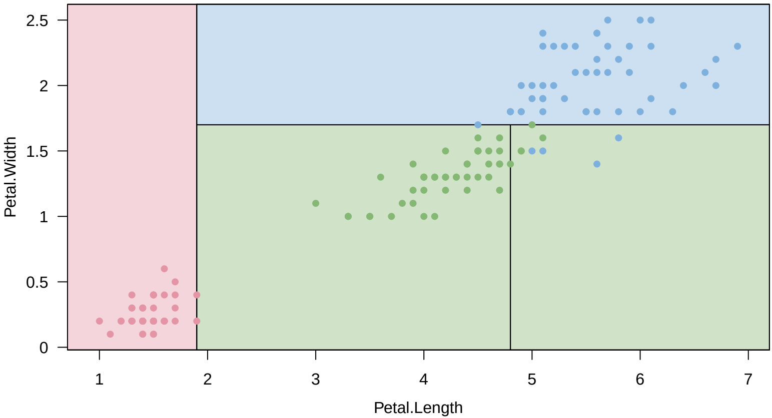
Tree Building Example V



Tree Building Example VI



Tree Building Example VII



Tree Building Algorithms

- AID (Sonquist and Morgan, 1964)
- CHAID (Kass, 1980)
- CART (Breiman et al., 1984) <– We mainly focus on this
  Classification and Regression Trees.
  Only builds binary trees.
- C4.5 (Quinlan, 1993)
- Unbiased Recursive Partitioning (Hothorn et al., 2006)

CART: Goodness of Fit I

- **Continuous targets:** Minimal SSE / variance
  Dividing all of the data with respect to the split variable X_j at splitpoint s, leads to the following half-spaces

  $$R_1(j,s) = \{X : X_j \leq s\} \text{ and } R_2(j,s) = \{X : X_j > s\}.$$

  Determination of the best split variable and the corresponding splitpoint:

  $$\min_{j,s}\left(\min_{c_1}\sum_{X_i\in R_1(j,s)}(Y_i - c_1)^2 + \min_{c_2}\sum_{X_i\in R_2(j,s)}(Y_i - c_2)^2\right).$$

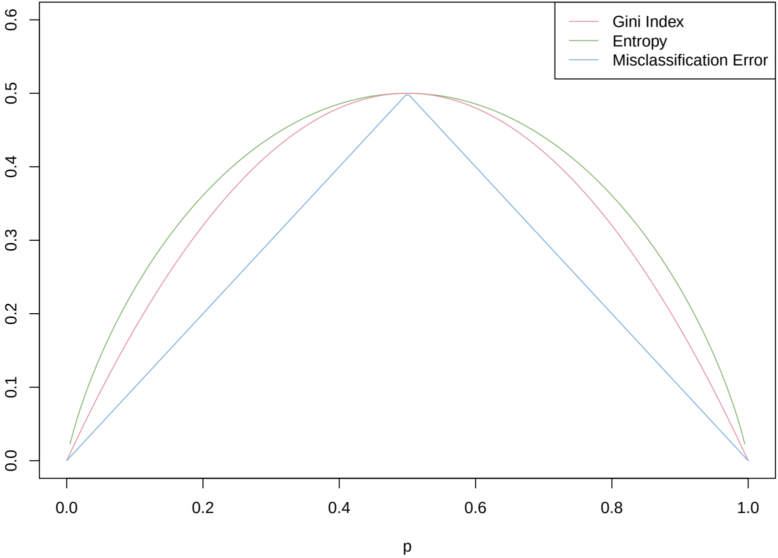
  for arbitrary j and s the inner minimization is solved through:
  $\hat{c}_1 = \text{mean}(Y_i|X_i \in R_1(j,s))$ and $\hat{c}_2 = \text{mean}(Y_i|X_i \in R_2(j,s))$

CART: Goodness of Fit II

- **Categorical targets (K categories):** "Impurity Measures"
  - Gini-Index:
  $$\sum_{k \neq k'} \hat{p}_k \hat{p}_{k'} = \sum_{k=1}^{K} \hat{p}_k (1 - \hat{p}_k)$$
  - Misclassification Error:
  $$1 - \arg\max_k \hat{p}_k$$
  - Entropy:
  $$-\sum_{k=1}^{K} \hat{p}_k \log \hat{p}_k \ ,$$

  where $\hat{p}_k$ corresponds to the relative frequency of category k

CART: Goodness of Fit III



CART: Stopping-Criteria

- Minimal number of observations per node, for a split to be tried
- Minimal increase in goodness of fit, for a split to be tried
- Minimal number of observations that must be contained in a leaf
- Maximal number of levels for tree

Advantages

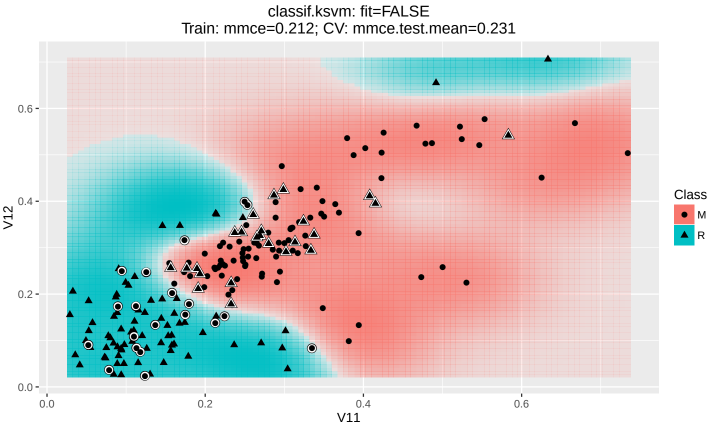
- Model is easy to comprehend
- Graphical visualizations allow good interpretability
- Interaction effects between features can be captured
- Tree structure reflects stepwise decisions
- Works for non-linear functions as well
- Built-in feature selection
- Can handle missing values
- Fast implementations exist for large data sizes
- Robust versus feature outliers or skewed feature distributions
- **Principle very flexible, custom trees for many tasks can be built**

Disadvantages

- High instability (variance) of the trees: Small changes in the data can potentially lead to completely different splits, and therefore to completely different trees as well
- Prediction function isn't smooth (a step function is fitted)
- Linear dependencies must be modeled over several splits, simple linear correlations must be translated into a complex tree structure
- Really not the best predictor. But we use trees to create forests and boosting models to achieve state-of-the-art performance!

mlr - Machine Learning in R

```
lrn = makeLearner("classif.ksvm")
plotLearnerPrediction(lrn, sonar.task, features = c("V11", "V12"))
```



`mlr` - Machine Learning in R

- `https://github.com/mlr-org/mlr`
- Clear interface to R classification, regression, clustering and survival analysis methods
- More than 100 “basic” ML algorithms! (not counting meta techniques)
- Fit, predict, evaluate and resample models
- Extensive visualizations for e.g. ROC curves, predictions and partial predictions
- Benchmarking of learners for multiple data sets
- Hyperparameter tuning, feature selection, pre-processing
- Combine different processing steps to a complex data mining chain that can be jointly optimized
- OpenML connector for the Open Machine Learning server
- Parallelization is built-in
- Detailed tutorial online