Introduction to Machine Learning

ML-Basics: Losses & Risk Minimization

HOW TO EVALUATE MODELS

OVERVIEW

No Free Lunch In machine learning, there's something called the "No Free Lunch" theorem. In a nutshell, it states that no one algorithm works best for every problem, and it's especially relevant for supervised learning (i.e. predictive modeling).
For example, you can't say that neural networks are always better than decision trees or vice-versa. There are many factors at play, such as the size and structure of your dataset.
As a result, you should try many different algorithms for your problem, while using a hold-out "test set" of data to evaluate performance and select the winner. Hypothesisspace + Risk + Optimization

CART FUNCTIONALITY

NON-PARAMETRIC | WHITE-BOX | FEATURE SELECTION

General idea Starting from a root node, ***classification & regression trees (CART)*** perform repeated **binary splits** of the data according to feature values, thereby subsequently dividing the input space $\mathcal{X}$ into M **rectangular partitions**.

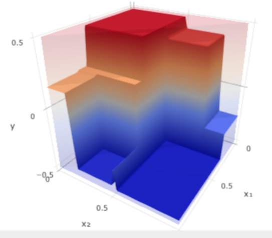
→ Pass observations along until each ends up in exactly one leaf node

→ In each step, find the optimal feature-threshold combination to split by

→ Assign response c_m to leaf node m

Hypothesis space

$$\mathcal{H} = \left\{ f(\mathbf{x}) : f(\mathbf{x}) = \sum_{m=1}^{M} c_m \mathbb{I}(\mathbf{x} \in Q_m) \right\}$$



Loss functions

Classification: mostly ***Brier score, Bernoulli loss***
Regression: mostly ***quadratic loss***

Optimization Exhaustive search for optimal splitting criterion (greedy optimization)

Hyperparameters Tree depth, minimum number of observations per node, ...

CART PRO'S & CON'S

Advantages

- + **Easy** to understand, interpret & visualize
- + Automatic handling of **non-numerical** features
- + Built-in **feature selection**
- + Automatic handling of **missings**
- + **Interaction** effects between features easily possible, even of higher orders
- + **Fast** computation and good scalability
- + High **flexibility** (custom split criteria or leaf-node prediction rules)

Disadvantages

- \- Rather **low accuracy** (at least, without bagging or boosting)
- \- High **variance/instability**: strong dependence on training data
- \- Therefore, poor generalization & risk of **overfitting**
- \- Several steps required for modeling linear relationships
- \- In presence of categorical features, **bias** towards features with **many categories**

Simple and good with feature selection, but not the best predictor

CART APPLICATION

For applications of CART, note the following:

Pruning / early stopping

Unless interrupted, splitting will go on until each leaf node contains a single observation (expensive + overfitting!)

$\rightarrow$ Use **pruning** and **stopping criteria** to limit complexity.

Implementation

R: package `rpart`
Python: `DecisionTreeClassifier` from package `scikit-learn`

Bagging

Since CART are instable predictors on their own, they are typically ensembled to form a **random forest**.