TWIX

Trees WItth eXtra splits

Sergej Potapov
Martin Theus
Simon Urbanek
Motivation

- Where the classical CART algorithm fails
  - Greedy algorithms never go for a (locally) second best solution, which would result in a better overall (global) solution.

Example: XOR-data
Trees: More Problems

- Non-orthogonal splitting directions ...

… can not be handled by single trees, no matter how we split
Bagging Revisited

Bagging = Bootstrap Aggregation, tries to simulate an infinite sample by bootstrapping, i.e. sampling from the original sample with replacement.

Repeat $N$ times:

1. Generate a bootstrap sample $D_i$ of size $n$.

2. Fit model $\hat{f}_{D_i}$.

Depending on the problem the $N$ results are aggregated:

- Classification: $g(x) = \arg\max_{c \in \mathcal{C}} \sum_{i=1}^{N} I(f_{D_i}(x) = c)$

- Regression: $g(x) = \frac{1}{N} \sum_{i=1}^{N} f_{D_i}(x)$
Ensembles

• **General Idea**
  Use many “different” classifier and combine them to get more accurate results.

• Bagging: Instability of trees yields different models

• Random Forests: Restrict input space randomly to get wider range of models

• Boosting: Iterate to down-weight “bad” points

**Question:**
Why use randomly generated (sub-optimal) models?
Tree Mechanics

- CART is a recursive partitioning algorithm
- Each node is split according to the maximum gain in the loss function
- Mountain plots shows the loss function for a variable for all possible split points
Idea behind TWIX

- Since the greedy CART algorithm not necessarily finds the “optimal” tree, try second best splits.
- Use these forests for bagging
- Expect better results for both single trees and aggregations

Problems

- How to find “good” candidates for second best splits?
- Number of inner nodes grows exponentially with the number of levels in the tree
  ⇒ so does the number of alternative trees
Second Best Splits: South African Heart Data
Second Best Splits: Global vs. Local

- When searching for a “best” split point, we can either look for
  - all top $n$ greatest deviance gains, or
  - only look for local maxima

- Example
  Top 6 splits
Second Best Splits: Forcing Variables

- Often a single variable dominates the potential deviance gain, and shadows all other variables
  → Many probably good split points are lost.

- Solution:
  Force a minimum number of split points for each variable.

- Example: top 6 vs. top 3
Second Best Splits: Grid Search

- In some situations good split points might not even associated with some (local) maximum in deviance gain.
  (Remember the XOR Example)

- Grid searches are most exhaustive, but also most expensive.
Implementation: The Grid

- If we allow $s_j$ splits per node on level $j$ of the tree, we get a maximum of
  \[ S = \prod_{i=1}^{k} s_i^{2^{i-1}} \]
  trees for a tree with no more than $k$ levels. Example:
  \[ s = (7, 4, 2) \Rightarrow S = 7^0 \cdot 4^1 \cdot 2^2 = 7 \cdot 16 \cdot 16 = 1792 \]
  ⇒ Work on a grid of computers
Implementation:
The R-Package

TWIX {TWIX}

Top Classification Trees

Description

Top Classification Trees

Usage

```r
TWIX(formula, data = NULL, test.data = NULL, subset = NULL,
      method = "deviance", topn.method = "complete", cluster = NULL,
      minsplit = 20, minbucket = round(minsplit/3), Devmin = 0.01,
      topN = 1, level = 6, st = 1, cl.level = 2, tol = 0.01, ...)
```

Arguments

- `formula`: formula of the form `y ~ x1 + x2 + ...`, where `y` must be a factor and `x1, x2, ...` are numeric.
- `data`: an optional data frame containing the variables in the model (training data).
- `test.data`: a data frame containing new data.
- `subset`: an optional vector specifying a subset of observations to be used.
- `method`: Which split points will be used? This can be "deviance" (default), "grid" or "local". If the method is set to "local" the program uses the local maxima of the split function (entropy), deviance all values of the entropy, grid grid points.
- `topn.method`: one of "complete" (default) or "single". A specification of the consideration of the split points. If set to "complete" it uses split points from all variables, else it uses split points per variable.
- `cluster`: name of the cluster, if parallel computing will be used.
- `minsplit`: the minimum number of observations that must exist in a node.
- `minbucket`: the minimum number of observations in any terminal <leaf> node.
- `Devmin`: the minimum improvement on entropy by splitting.
- `topN`: integer vector. How many splits will be selected and at which level? If length 1, the same size of splits will be selected at each level. If length > 1, for example `topN=c(3,2)`, 3 splits will be chosen at first level, 2 splits at second level and for all next levels 1 split.
- `level`: maximum depth of the trees. If level set to 1, trees consist of root node.
- `st`: step parameter for method "grid".
- `cl.level`: parameter for parallel computing.
- `tol`: parameter, which will be used, if `topn.method` is set to "single".
- `...`: further arguments to be passed to or from methods.
“Driving the Beast”

- **The most important tuning parameters are**
  - **method**
    - Which split points will be used? This can be "deviance" (default), "grid" or "local". If the method is set to: *local* the program uses the local maxima of the split function (entropy), *deviance* all values of the entropy, *grid* grid points.
  - **topn.method**
    - one of "complete" (default) or "single". A specification of the consideration of the split points. If set to "complete" it uses split points from all variables, else it uses split points per variable.
  - **topN**
    - integer vector. How many splits will be selected and at which level? If length 1, the same size of splits will be selected at each level. If length > 1, for example topN=c(3,2), 3 splits will be chosen at first level, 2 splits at second level and for all next levels 1 split.
  - **level**
    - maximum depth of the trees. If level set to 1, trees consist of root node.
  - **Stopping Rules:**
    - **minsplit**
      - the minimum number of observations that must exist in a node.
    - **minbucket**
      - the minimum number of observations in any terminal <leaf> node.
    - **Devmin**
      - the minimum improvement on entropy by splitting.
South African Heart Disease Data cont.

• To get a “fair”, i.e. generalizable and not too overfitted classifier, we usually split the data into 3 chunks:

- **Training**
  All models are trained using the training data

- **Validation**
  The “best” model is selected using the validation data
  (The chosen model is then estimated with training+validation)

- **Test**
  The performance is then assessed with the test data

What about trees?

Model Structure = Model parameters
The Dataset

- 10 Variables, 462 Observations
- Target: **Coronary Heart Disease (chd)**, 34.63% = 160 cases
- Inputs:
  - continuous
    - sbp: systolic blood pressure
    - tobacco: cumulative tobacco (kg)
    - ldl: low density lipoprotein cholesterol
    - adiposity
    - typea: type-A behavior
    - obesity
    - alcohol: current alcohol consumption
    - age: age at onset
  - discrete
    - famhist: family history of heart disease (Present, Absent)
The Dataset: Univariate

sbp

1.0

0.0

101 218
The Dataset: Univariate
The Dataset: Bivariate
The Dataset: Multivariate
The Competitors: On 100 random samples

- **Logistic Regression**
  
  \[
  \text{glm(response~., data=dataTrain, family="binomial")}
  \]

- **Traditional CART**
  
  \[
  \text{rpart(response ~ ., data=dataTrain,}
  
  \text{parms=list(split='information'))}
  \]

- **Bagging**
  
  \[
  \text{bagging(response~., data=dataTrain, nbagg=100)}
  \]

- **SVM**
  
  \[
  \text{svm(response~., data=dataTrain)}
  \]

  !! None of the methods has been further tuned !!
Competitors Results: Error Rates

- Trad. Trees
- Bagging
- Support Vektor Machines
- Log. Regression
TWIX: Diagnostics

- For a given “Multitree” we can compare deviance and classification rate on training and test/validation data.

Example:
TWIX: Tree Selection

- The CCR (Correct Classification Rate) of the top TWIX trees are far better than those of greedy trees and most other classification methods

**Quest:**
How to find the “best” trees from the validation data?

- Currently:
  
  Sort trees according to
  - training deviance
  - test deviance
  - training CCR
  - test CCR

  and pick the best!
TWIX: Results

• Local Maxima across all variables (50 samples)
TWIX: Results

- Local Maxima, within all variables (50 samples)
TWIX: Results

- 100 runs of a (10,5,2) tree
Bagged TWIX

• What bagging should look like:
Bagged TWIX: Example

- Bagging TWIX trees does often not improve the CCR.
Conclusion

• For the South-African-Heart Data
  – Single TWIX-trees out-perform traditional tree and usually bagged trees
  – Bagged TWIX beats bagged trees and reaches top performance
  – TWIX gives good single alternative tree models

• Still much room for performance improvement
  – Stopping rules and pruning
  – Better tree selection
  – Improved selection of second best splits
  – More tests on more datasets
  – Better understanding of the tree-families

• Computational effort is high, but parallel computing speeds up dramatically

• Complex methods are hard to implement and hard to test