Resampling

Fabrice Rossi

SAMM
Université Paris 1 Panthéon Sorbonne

2019

A major ML difficulty

Standard supervised learning setting

- $\mathcal{D} = ((\mathbf{X}_i, \mathbf{Y}_i))_{1 \leq i \leq N}$
- $(\mathbf{X}_i, \mathbf{Y}_i) \sim D$ and $\mathcal{D} \sim D^N$ (product distribution)
- l: loss function
- ML goal: from $\mathcal{D}$ find g such that $R_l(g) = \mathbb{E}_{(\mathbf{X},\mathbf{Y})\sim D}(l(g(\mathbf{X}), \mathbf{Y}))$ is minimal

Empirical risk

- D is unknown and $R_l(g)$ cannot be computed
- $\widehat{R}_l(g, \mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} l(g(\mathbf{X}_i), \mathbf{Y}_i)$
- if g is obtained from $\mathcal{D}$, $\widehat{R}_l(g, \mathcal{D})$ can strongly **underestimate** $R_l(g)$

Example

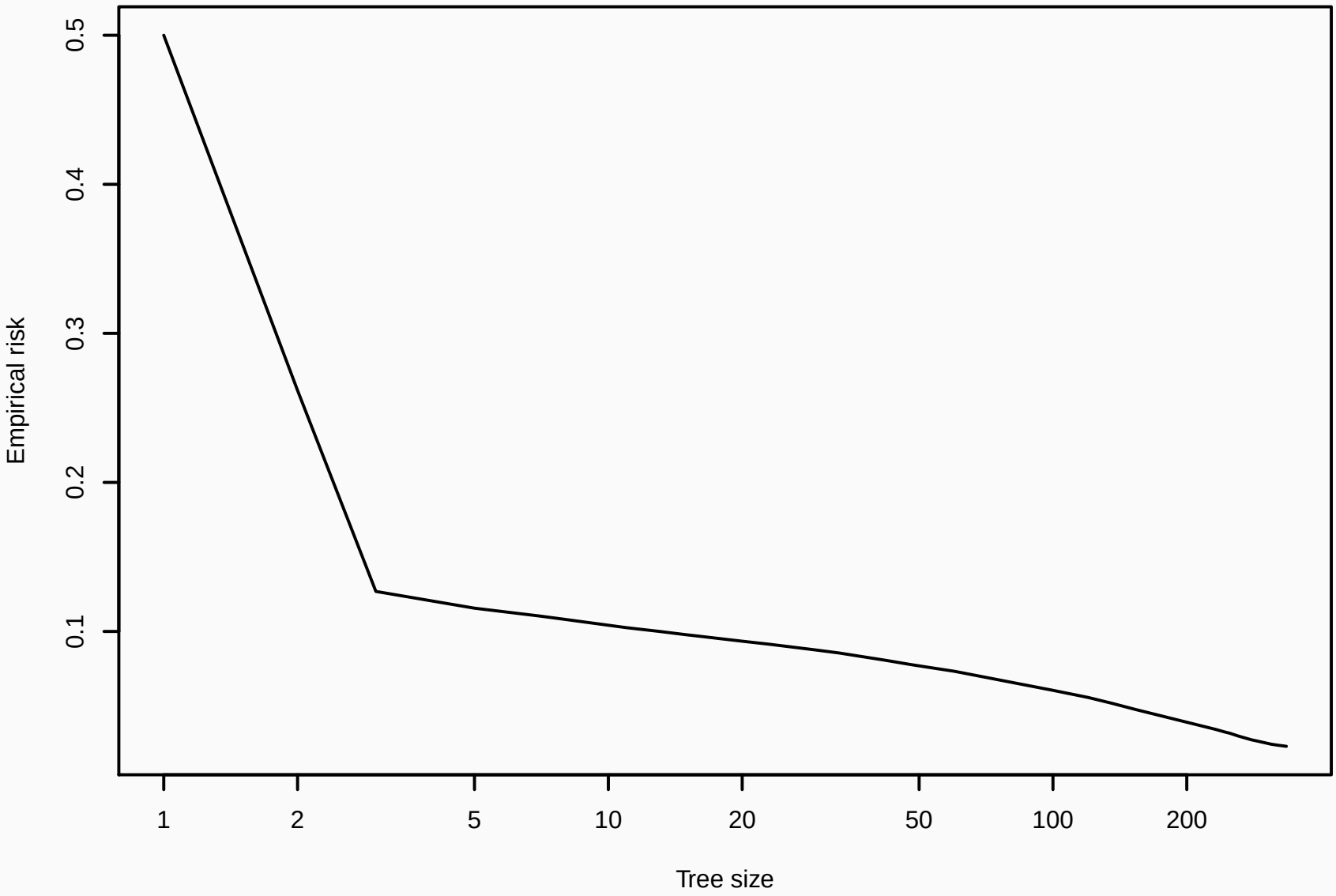
Data: Bank+Marketing

- direct marketing campaign (phone calls)
- 20 variables: client related (e.g. age), context related (e.g. eurobir rate), campaign related (last call, etc.)
- target variable: will the client subscribe a product?
- artificially balanced in this presentation (6496 training examples)

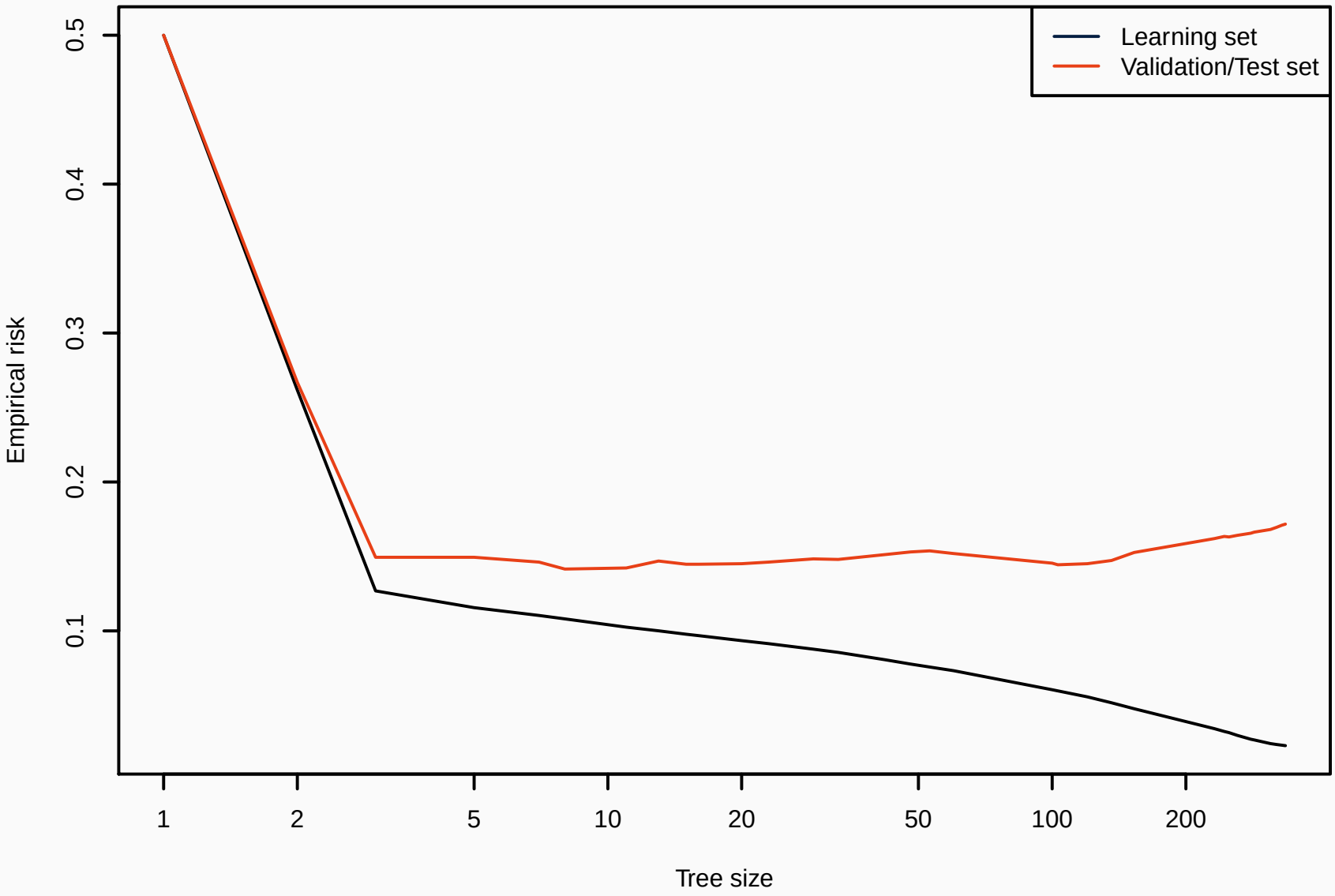
Decision tree

- stopping criteria: $\min_{size} = 10$ and $\min_{improv} = 0$
- standard binary loss

Example



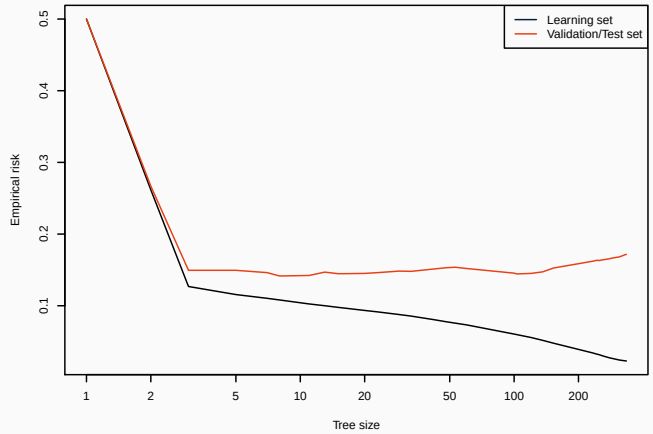
Example


Learning set
Validation/Test set
Empirical risk
0.1
0.2
0.3
0.4
0.5
1
2
5
10
20
50
100
200
Tree size

Typical behavior

First phase

- (fast) decrease of the empirical risk on both sets
- limited gap between empirical risks
- proper learning: from underfitting to the optimal model



Plateau phase

- continued decrease on the learning set
- increasing gap between empirical risks
- no progress on the validation set

Overfitting phase

- continued decrease on the learning set
- increase of the empirical risk on the validation set
- noise learning

Basic general framework

General algorithm

1. split the data into $\mathcal{D}$ (learning), $\mathcal{D}'$ (validation) and $\mathcal{D}''$ (test)
2. for each machine learning algorithm $\mathcal{A}$ under study
   - 2.1 for each value θ of the parameters of the algorithm
     - 2.1.1 compute the model using θ on $\mathcal{D}$, $g_{\mathcal{A},\theta,\mathcal{D}}$
     - 2.1.2 compute $\widehat{R}_l(g_{\mathcal{A},\theta,\mathcal{D}}, \mathcal{D}')$
3. chose the best algorithm with the best parameter, $\mathcal{A}^*$ and θ^* (according to $\widehat{R}_l(., \mathcal{D}')$)
4. compute the best model $g^* = g_{\mathcal{A}^*,\theta^*,\mathcal{D}\cup\mathcal{D}'}$
5. compute $\widehat{R}_l(g^*, \mathcal{D}'')$

Two levels procedure

- in both case: learning + evaluation
- similar solutions can be used

Other frameworks?

Splitting is limited

- a lot of data is needed!
- parameters: size compromise?
- randomization effects
- both aspects are limited when the data set is huge

Other solutions

- for smaller data sets
- roots in resampling techniques in statistics (jackknife and bootstrap)
- two principles:
  1. replacing richness in data by richness in computational resources
  2. resampling: simulating “new” data sets from the original one

Cross-validation

The standard principle

- a.k.a. out-of-sample testing: a learning set and an independent evaluation set
- evaluation: validation or test
- theoretical justification: strong law of large numbers, if $g \perp\!\!\!\perp \mathcal{D}'$

$$\widehat{R}_l(g, \mathcal{D}') \xrightarrow[|\mathcal{D}'|\to\infty]{a.s.} R_l(g)$$

Cross-validation

The standard principle

- a.k.a. out-of-sample testing: a learning set and an independent evaluation set
- evaluation: validation or test
- theoretical justification: strong law of large numbers, if $g \perp\!\!\!\perp \mathcal{D}'$

$$\widehat{R}_l(g, \mathcal{D}') \xrightarrow[|\mathcal{D}'|\to\infty]{a.s.} R_l(g)$$

In practice

- numerous variants
- degenerate case of *simple validation*: random split sample

Leave-one-out

Principle

- exhaustive cross-validation
- for each $(\mathbf{X}_i, \mathbf{Y}_i) \in \mathcal{D}$:
  - learn a model on $\mathcal{D} \setminus \{(\mathbf{X}_i, \mathbf{Y}_i)\}$
  - test it on $\mathcal{D}' = \{(\mathbf{X}_i, \mathbf{Y}_i)\}$
- average the losses to get an estimate of the risk

The Leave-one-out risk estimate

- $g = A(((\mathbf{X}_i, \mathbf{Y}_i))_{1 \leq i \leq N})$ where A is a machine learning algorithm with its parameters
- $g_{-k} = A((\mathbf{X}_1, \mathbf{Y}_1), \ldots, (\mathbf{X}_{k-1}, \mathbf{Y}_{k-1}), (\mathbf{X}_{k+1}, \mathbf{Y}_{k+1}), \ldots, (\mathbf{X}_N, \mathbf{Y}_N))$
- risk estimate $\hat{R}_l(g, \mathcal{D})_{loo} = \dfrac{1}{N} \displaystyle\sum_{k=1}^{N} l(g_{-k}(\mathbf{X}_k), \mathbf{Y}_k)$

Pros and cons

Pros

- $\hat{R}_l(g, \mathcal{D})_{loo}$ has generally a low bias
- no randomness and no parameter
- equal use for each data point
- obvious parallel implementation

Cons

- $\hat{R}_l(g, \mathcal{D})_{loo}$ has generally a high variance
- very large computational load

Bias and variance

Estimator

- $\hat{R}_l(g, \mathcal{D})_{loo}$ is an estimator of $R_l(g)$
- the variability in the estimation is induced by the data set
- bias: $\mathbb{E}_{\mathcal{D} \sim D^N} \left(\hat{R}_l(g, \mathcal{D})_{loo} \right) - R_l(g)$
- variance: $\mathbb{V}_{\mathcal{D} \sim D^N} \left(\hat{R}_l(g, \mathcal{D})_{loo} \right)$

Sources of bias and variance

- bias:
  - g is incorrectly estimated
  - on a subset of $\mathcal{D}$ for instance
- variance:
  - $l(g(\mathbf{X}), \mathbf{Y})$ is “unstable”
  - not enough estimation points

Fast cases

Computational cost

- each g_{-k} is learned on almost all the data
- the cost is frequently too large to big realistic
- but there are fast cases

Linear models

- all target values: $\mathbb{Y} = (\mathbf{Y}_1, \ldots, \mathbf{Y}_N)^T$
- all predicted values: $\hat{\mathbb{Y}}$
- in some models there is matrix $\mathbf{S}$ such that $\hat{\mathbb{Y}} = \mathbf{S}\mathbb{Y}$
- then in general for the quadratic loss

$$\hat{R}(g, \mathcal{D})_{loo} = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{\mathbf{Y}_i - \hat{\mathbb{Y}}_i}{1 - \mathbf{S}_{ii}} \right)^2$$

Linear regression

Standard results

- linear model $g_{\beta_0,\boldsymbol{\beta}}(\mathbf{x}) = \beta_0 + \boldsymbol{\beta}^T\mathbf{x}$
- optimal solution $\left(\beta_0^*, \boldsymbol{\beta}^*\right)^T = \left(\mathbb{X}^T\mathbb{X}\right)^{-1}\mathbb{X}^T\mathbb{Y}$ where

$$\mathbb{X} = \begin{pmatrix} 1 & \mathbf{X}_1^T \\ \dots & \dots \\ 1 & \mathbf{X}_N^T \end{pmatrix}$$

Leave-one-out

- then $\hat{\mathbb{Y}} = \underbrace{\mathbb{X}\left(\mathbb{X}^T\mathbb{X}\right)^{-1}\mathbb{X}^T}_{\mathbf{S}}\mathbb{Y}$
- and the equation applies $\hat{R}(g,\mathcal{D})_{loo} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{\mathbf{Y}_i - \hat{\mathbb{Y}}_i}{1-\mathbf{S}_{ii}}\right)^2$

In practice

Linear model

- almost no overhead: $\hat{R}(g, \mathcal{D})_{loo}$ is obtained as a consequence of fitting the model
- can be use for
  - feature selection in linear models
  - parameter optimization in ridge regression
- straightforward extension to nonlinear models such as the kernel ridge regression

Other models

- leave-one-out is generally too costly
- excepted with tiny data sets such as some medical ones

K-fold cross-validation

Goals

- variance reduction: larger evaluation data set
- computational load reduction: less models
- symmetry: equal use for each data point

Fold based solution

- random partition of $\{1, \ldots, N\}$ into K folds, $C_1, \ldots, C_K$
- $\kappa(i)$: fold of (X_i, Y_i), i.e. $i \in C_{\kappa(i)}$
- g_{-k}: model estimated on $\mathcal{D}_{-k} = \{(\mathbf{X}_i, \mathbf{Y}_i) \in \mathcal{D} \mid \kappa(i) \neq k\}$
- K-fold cross-validated risk estimate

$$\hat{R}_l(g, \mathcal{D})_{cv} = \frac{1}{N} \sum_{i=1}^{N} l(g_{-\kappa(i)}(\mathbf{X}_i), \mathbf{Y}_i)$$

Consequences

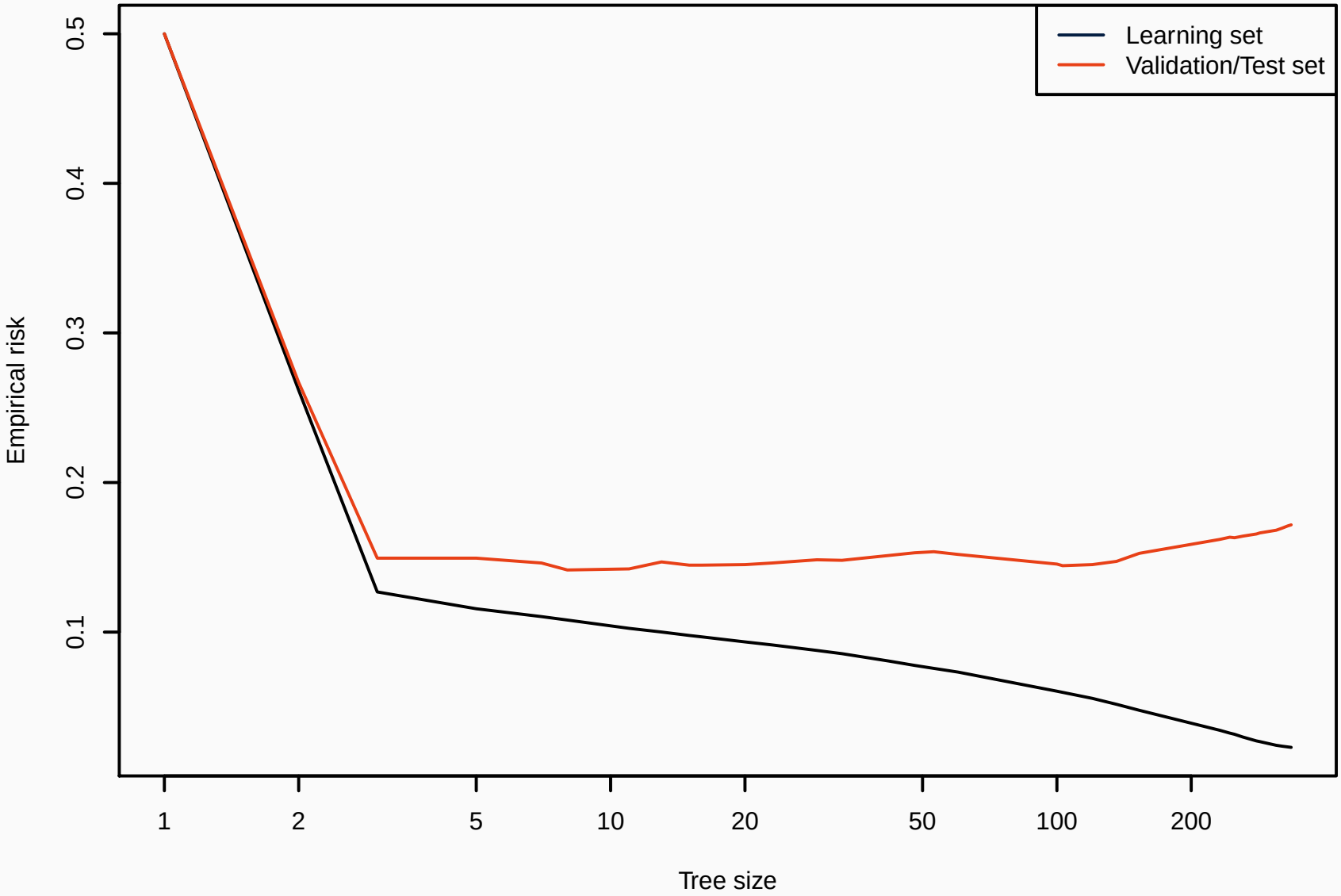
Compromise

- large K:
  - large computational load
  - small bias
  - large variance
- standard comprise: between 5 and 10 folds

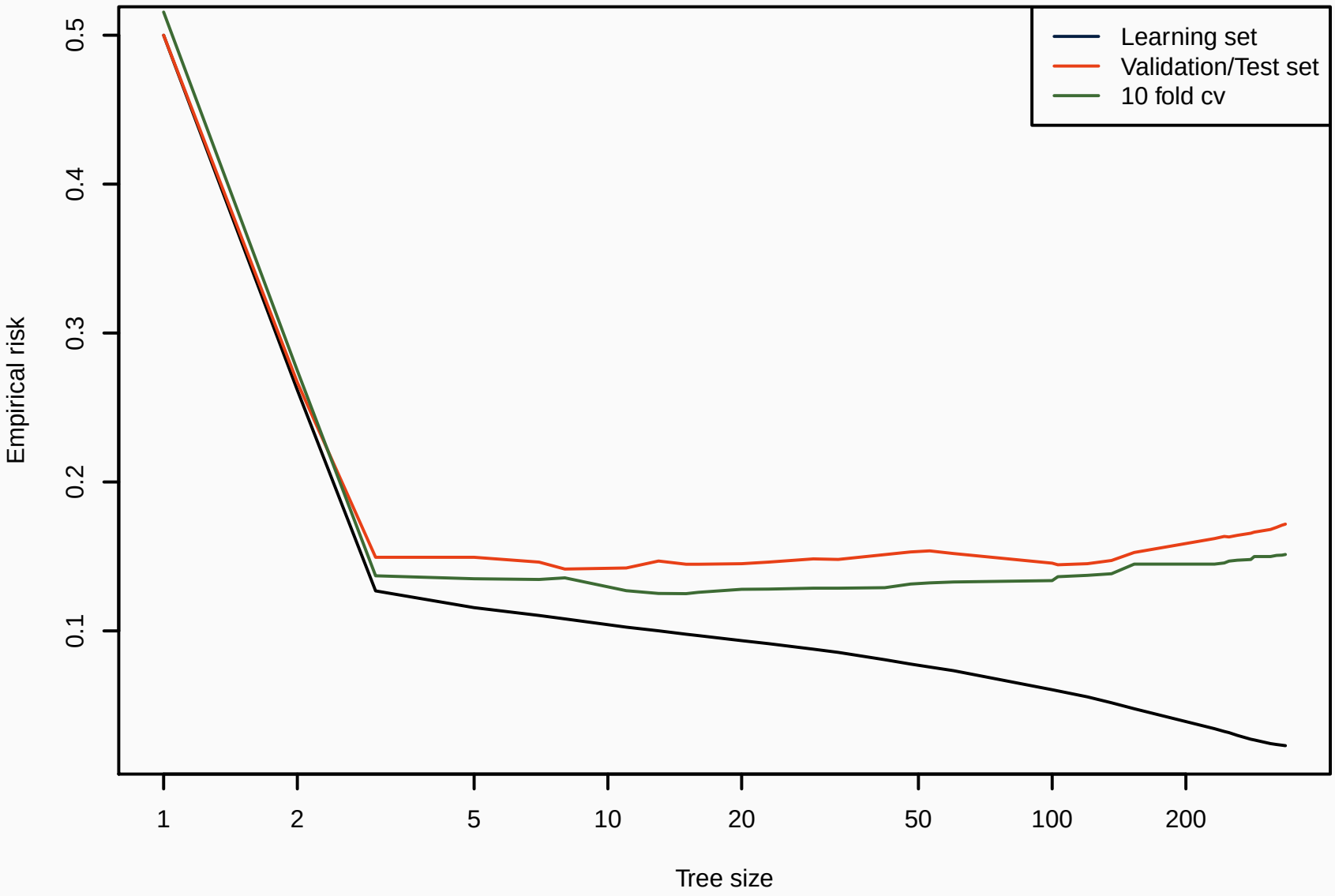
Problems

- no fast version
- effects of the value of K?
- effects of the random partition?

Example


Learning set
Validation/Test set
Empirical risk
0.1
0.2
0.3
0.4
0.5
1
2
5
10
20
50
100
200
Tree size

Example


Learning set
Validation/Test set
10 fold cv
Empirical risk
0.1
0.2
0.3
0.4
0.5
1
2
5
10
20
50
100
200
Tree size

Stratified CV

Issue

- folds are random
- the data distribution in a fold could be too different from the global distribution
- stronger potential issue for unbalanced data

Stratified sampling

- make sure that the target variable has the same distribution in each fold
- quantile based for continuous target variable
- partition technique:
  - split into subsets with similar values for **Y**
  - build random partitions for each subset
  - merge the folds

Randomized CV

Goals

- decoupling of K and of the proportion of learning/evaluation examples
- reducing the impact of the random partitioning

Method

- two parameters: p (sampling ratio) and B repetition number
- build B independent random subsets of $\{1, \ldots, N\}$, $A_1, \ldots, A_B$ with $|A_k| = p \times N$
- g_k is the model learned on A_k
- repeated cross-validated risk estimate

$$\hat{R}_l(g, \mathcal{D})_{rcv} = \frac{1}{B} \sum_{k=1}^{B} \frac{1}{(1-p) \times N} \sum_{i \notin A_k} l(g_k(\mathbf{X}_i), \mathbf{Y}_i)$$

Practical aspects

Parameters

- large B:
  - slow
  - reduced variance
- p:
  - fairly similar role as the one of k in standard K fold CV
  - classical values between 0.5 and 0.8

Other aspects

- should be stratified
- very high natural variance compared to standard K fold CV

Comparing cross-validated estimates

Variance

- estimators of the form $\hat{R}_l(g, \mathcal{D})_*$ have generally a high variance
- can we compare e.g. $\hat{R}_l(g_1, \mathcal{D})_{cv}$ and $\hat{R}_l(g_2, \mathcal{D})_{cv}$?

Comparing cross-validated estimates

Variance

- estimators of the form $\hat{R}_l(g, \mathcal{D})_*$ have generally a high variance
- can we compare e.g. $\hat{R}_l(g_1, \mathcal{D})_{cv}$ and $\hat{R}_l(g_2, \mathcal{D})_{cv}$?
- conservative answer: no!

Comparing cross-validated estimates

Variance

- estimators of the form $\hat{R}_l(g, \mathcal{D})_*$ have generally a high variance
- can we compare e.g. $\hat{R}_l(g_1, \mathcal{D})_{cv}$ and $\hat{R}_l(g_2, \mathcal{D})_{cv}$?
- conservative answer: no!

Solutions

- test based approach:
  - run many cross-validations $\hat{R}_l(g, \mathcal{D})_{cv,1}, \ldots, \hat{R}_l(g, \mathcal{D})_{cv,p}$
  - use paired Wilcoxon test to decide whether a model is better than another
- or estimate the variance of a cv by $\hat{\mathbb{V}}(\hat{R}_l(g, \mathcal{D})_{cv}) = \frac{1}{N} \sum_{i=1}^{N} (l(g_{-\kappa(i)}(\mathbf{X}_i), \mathbf{Y}_i) - \hat{R}_l(g, \mathcal{D})_{cv})^2$
- or at least use the same random sources for all models (e.g. same folds)

Bootstrap

Presentation

- the bootstrap is a statistical method proposed by Efron in 1979
- it is based on *bootstrap samples*:
  - sampling distribution: uniform in the original sample
  - sample size: identical to the original one
- bootstrap samples are used as approximate samples from the original unknown distribution

Example: variance of an estimator

- a standard estimator: $\hat{\theta} = f(X_1, \ldots, X_N) = f(\mathcal{D})$
- generate B bootstrap samples (at least 50) from $\mathcal{D}$, $\mathcal{D}^1$ to $\mathcal{D}^B$
- estimate the variance of the estimator using

$$\widehat{\sigma^2}_{boot}(\hat{\theta}) = \frac{1}{B-1} \sum_{b=1}^{B} \left(\hat{\theta}(\mathcal{D}^b) - \hat{\theta}_{boot}(\mathcal{D}) \right)^2 \text{ with } \hat{\theta}_{boot}(\mathcal{D}) = \frac{1}{B} \sum_{b=1}^{B} \hat{\theta}(\mathcal{D}^b)$$

Application to machine learning

Bias estimation (Efron 1979 [Efr79])

- estimator under study: $\hat{\theta} = R_l(g) - \hat{R}_l(g, \mathcal{D})$
- bootstrap is used to estimate the expectation of $\hat{\theta}$ with

$$\hat{\theta}_{boot}(\mathcal{D}) = \frac{1}{B} \sum_{b=1}^{B} \left(\hat{R}_l(g_b, \mathcal{D}) - \hat{R}_l(g_b, \mathcal{D}^b) \right),$$

  where g_b is the model learned on $\mathcal{D}^b$ the bootstrap sample
- adjusted empirical risk: $\hat{R}_l(g, \mathcal{D})_{boot} = \hat{R}_l(g, \mathcal{D}) + \hat{\theta}_{boot}(\mathcal{D})$

Application to machine learning

Direct estimation: *leave-one-out bootstrap* (Efron 1983 [Efr83])

- comparable to repeated randomized cross-validation
- $\mathcal{D}^b$ contains on average $63,2$ % of $\mathcal{D}$
- one can estimate the risk on $\overline{\mathcal{D}^b}$ (roughly $36,8$ % of the original set)

$$\hat{R}_l(g, \mathcal{D})_{loob} = \frac{1}{B} \sum_{b=1}^{B} \frac{1}{|\overline{\mathcal{D}^b}|} \sum_{i \in \overline{\mathcal{D}^b}} l(g_b(X_i), Y_i)$$

- large bias (comparable to 2-folds or 3-folds cross-validation)

Bias correction techniques

Bootstrap .632 (Efron 1983 [Efr83])

- *leave-one-out bootstrap* overestimates the true risk
- can be compensated using the empirical risk (which underestimates the true risk!)
- .632 estimator

$$\hat{R}_l(g, \mathcal{D})_{.632} = 0,368 \times \hat{R}_l(g, \mathcal{D}) + 0,623 \times \hat{R}_l(g, \mathcal{D})_{loob}$$

Bias correction techniques

Bootstrap .632+ (Efron & Tibshirani 1997 [ET97])

- improvement over the Bootstrap .632 in case of massive overfitting
- information less error rate

$$\hat{\gamma} = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} I(g(X_i), Y_j)$$

- relative over-fitting rate

$$\hat{R} = \frac{\hat{R}_I(g, \mathcal{D})_{loob} - \hat{R}_I(g, \mathcal{D})}{\hat{\gamma} - \hat{R}_I(g, \mathcal{D})}$$

- .632+ estimator

$$\hat{L}_{.632+}(g) = \frac{0,368 \times (1 - \hat{R}) \hat{R}_I(g, \mathcal{D}) + 0,632 \times \hat{R}_I(g, \mathcal{D})_{loob}}{1 - 0,368 \times \hat{R}}$$

Bootstrap in practice

Pros

- large statistical literature with numerous variants and improvements
- not specific at all to machine learning
- bias correction techniques are useful and lead to reduced bias (compared to cross-validation)
- duplicated examples can be handled efficiently in some algorithms

Cons

- slow!
- large variance unless B is large
- duplicate examples can be problematic in some algorithms

Take home message

Main message

Always use a validation based approach to estimate the risk of a model

In addition

- use at least K-fold cross-validation
- stratify the samples
- limit the effects of randomness by sharing the random parts

References I

B. Efron.

Bootstrap methods: Another look at the jackknife.
Ann. Statist., 7(1):1–26, 01 1979.

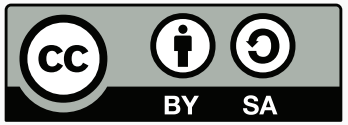
Bradley Efron.

Estimating the error rate of a prediction rule: Improvement on cross-validation.
Journal of the American Statistical Association, 78(382):316–331, 1983.

Bradley Efron and Robert Tibshirani.

Improvements on cross-validation: The .632+ bootstrap method.
Journal of the American Statistical Association, 92(438):548–560, 1997.

Licence



Version

Last git commit: 2019-01-17
By: Fabrice Rossi (Fabrice.Rossi@apiacoa.org)
Git hash: 21a7fea6a63ad78a37f024769a5abf0bebccd946