Local tree methods for classification

Jean-Michel Marin
Alice Cleynen and Louis Raynal

University of Montpellier

Bayesian statistics in the big data area
30th November 2018

Introduction

Complex statistical models $\Rightarrow$ **Intractable likelihood**

1. $f(\mathbf{y}|\theta) = \int f(\mathbf{y}, \mathbf{u}|\theta)\mu(\mathrm{d}\mathbf{u})$ intractable
   population genetics models, coalescent process

   EM algorithms, Gibbs sampling, pseudo-marginal MCMC methods, variational approximations

2. $f(\mathbf{y}|\theta) = g(\mathbf{y}, \theta)/Z(\theta)$ and $Z(\theta)$ intractable
   Markov random field

   pseudo-marginal MCMC methods, variational approximations

Introduction

Complex statistical models ⇒ **Intractable likelihood**

1. $f(\mathbf{y}|\boldsymbol{\theta}) = \int f(\mathbf{y}, \mathbf{u}|\boldsymbol{\theta})\mu(\mathrm{d}\mathbf{u})$ intractable
   population genetics models, coalescent process

   EM algorithms, Gibbs sampling, pseudo-marginal MCMC methods, variational approximations

2. $f(\mathbf{y}|\boldsymbol{\theta}) = g(\mathbf{y}, \boldsymbol{\theta})/Z(\boldsymbol{\theta})$ and $Z(\boldsymbol{\theta})$ intractable
   Markov random field

   pseudo-marginal MCMC methods, variational approximations

Introduction

Complex statistical models ⇒ **Intractable likelihood**

1. $f(\mathbf{y}|\boldsymbol{\theta}) = \int f(\mathbf{y}, \mathbf{u}|\boldsymbol{\theta})\mu(\mathrm{d}\mathbf{u})$ intractable
   population genetics models, coalescent process

   EM algorithms, Gibbs sampling, pseudo-marginal MCMC methods, variational approximations

2. $f(\mathbf{y}|\boldsymbol{\theta}) = g(\mathbf{y}, \boldsymbol{\theta})/Z(\boldsymbol{\theta})$ and $Z(\boldsymbol{\theta})$ intractable
   Markov random field

   pseudo-marginal MCMC methods, variational approximations

Introduction

Approximate Bayesian Computation (ABC)

Simulate pseudo-datasets using the Bayesian generative models and compare them to the observed dataset

Since the Likelihood-free rejection sampler for paramerter inference
Tavare et al. (1997) Genetics
Pritchard et al. (1999) Molecular Biology and Evolution
numerous developments:

1. regression adjustments,
2. more efficient algorithms,
3. model choice,
4. selection of the summary statistics...

Introduction

Approximate Bayesian Computation (ABC)

Simulate pseudo-datasets using the Bayesian generative models and compare them to the observed dataset

Since the Likelihood-free rejection sampler for paramerter inference
Tavare et al. (1997) Genetics
Pritchard et al. (1999) Molecular Biology and Evolution
numerous developments:

1. regression adjustments,
2. more efficient algorithms,
3. model choice,
4. selection of the summary statistics...

Introduction

Approximate Bayesian Computation (ABC)

Simulate pseudo-datasets using the Bayesian generative models and compare them to the observed dataset

Since the Likelihood-free rejection sampler for paramerter inference
Tavare et al. (1997) Genetics
Pritchard et al. (1999) Molecular Biology and Evolution
numerous developments:

1. regression adjustments,
2. more efficient algorithms,
3. model choice,
4. selection of the summary statistics...

Introduction

Approximate Bayesian Computation (ABC)

Simulate pseudo-datasets using the Bayesian generative models and compare them to the observed dataset

Since the Likelihood-free rejection sampler for paramerter inference
Tavare et al. (1997) Genetics
Pritchard et al. (1999) Molecular Biology and Evolution
numerous developments:

1. regression adjustments,
2. more efficient algorithms,
3. model choice,
4. selection of the summary statistics...

Introduction

Approximate Bayesian Computation (ABC)

Simulate pseudo-datasets using the Bayesian generative models and compare them to the observed dataset

Since the Likelihood-free rejection sampler for paramerter inference
Tavare et al. (1997) Genetics
Pritchard et al. (1999) Molecular Biology and Evolution
numerous developments:

1. regression adjustments,
2. more efficient algorithms,
3. model choice,
4. selection of the summary statistics...

Introduction

Approximate Bayesian Computation (ABC)

Simulate pseudo-datasets using the Bayesian generative models and compare them to the observed dataset

Since the Likelihood-free rejection sampler for paramerter inference
Tavare et al. (1997) Genetics
Pritchard et al. (1999) Molecular Biology and Evolution
numerous developments:

1. regression adjustments,
2. more efficient algorithms,
3. model choice,
4. selection of the summary statistics...

Introduction

Approximate Bayesian Computation (ABC)

Simulate pseudo-datasets using the Bayesian generative models and compare them to the observed dataset

Since the Likelihood-free rejection sampler for paramerter inference
Tavare et al. (1997) Genetics
Pritchard et al. (1999) Molecular Biology and Evolution
numerous developments:

1. regression adjustments,
2. more efficient algorithms,
3. model choice,
4. selection of the summary statistics...

Introduction

Most popular and efficient ABC strategy: **use of Machine Learning tools on the training set produced with the Bayesian generative model** (*the reference table*)

Fast e-free Inference of Simulation Models with Bayesian Conditional Density Estimation
Papamakarios and Murray (2016) NIPS

Approximate the whole posterior distribution by using Mixture Density Networks (Bishop, 1994) - Gaussian mixture models with parameters calibrated thanks to neural networks

The number of mixture components and the number of hidden layers of the networks require calibration

Introduction

Most popular and efficient ABC strategy: **use of Machine Learning tools on the training set produced with the Bayesian generative model** (*the reference table*)

Fast e-free Inference of Simulation Models with Bayesian Conditional Density Estimation
Papamakarios and Murray (2016) NIPS

Approximate the whole posterior distribution by using Mixture Density Networks (Bishop, 1994) - Gaussian mixture models with parameters calibrated thanks to neural networks

The number of mixture components and the number of hidden layers of the networks require calibration

Introduction

Most popular and efficient ABC strategy: **use of Machine Learning tools on the training set produced with the Bayesian generative model** (*the reference table*)

Fast e-free Inference of Simulation Models with Bayesian Conditional Density Estimation
Papamakarios and Murray (2016) NIPS

Approximate the whole posterior distribution by using Mixture Density Networks (Bishop, 1994) - Gaussian mixture models with parameters calibrated thanks to neural networks

The number of mixture components and the number of hidden layers of the networks require calibration

Introduction

Most popular and efficient ABC strategy: **use of Machine Learning tools on the training set produced with the Bayesian generative model** (*the reference table*)

Fast e-free Inference of Simulation Models with Bayesian Conditional Density Estimation
Papamakarios and Murray (2016) NIPS

Approximate the whole posterior distribution by using Mixture Density Networks (Bishop, 1994) - Gaussian mixture models with parameters calibrated thanks to neural networks

The number of mixture components and the number of hidden layers of the networks require calibration

Introduction

Deep Learning for Population Genetic Inference
Sheehan and Song (2016) PLOS Computational Biology

Deep learning makes use of multilayer neural networks to learn a feature-based function from the input (hundreds of correlated summary statistics) to the output (population genetic parameters of interest)

Unsupervised pretraining using autoencoders very interesting, but requires a lot of calibration

Introduction

Deep Learning for Population Genetic Inference
Sheehan and Song (2016) PLOS Computational Biology

Deep learning makes use of multilayer neural networks to learn a feature-based function from the input (hundreds of correlated summary statistics) to the output (population genetic parameters of interest)

Unsupervised pretraining using autoencoders very interesting, but requires a lot of calibration

Introduction

Deep Learning for Population Genetic Inference
Sheehan and Song (2016) PLOS Computational Biology

Deep learning makes use of multilayer neural networks to learn a feature-based function from the input (hundreds of correlated summary statistics) to the output (population genetic parameters of interest)

Unsupervised pretraining using autoencoders very interesting, but requires a lot of calibration

Introduction

Reliable ABC model choice via random forests
Pudlo, Marin et al. (2016) Bioinformatics

Deciphering the Routes of invasion of Drosophila suzukii by Means of ABC Random Forest
Fraimout et al. (2017) Molecular Biology and Evolution

ABC random forests for Bayesian parameter inference
Raynal, Marin et al. (2018) Bioinformatics

Learn on a huge reference table using Random Forests (RF)

Introduction

Reliable ABC model choice via random forests
Pudlo, Marin et al. (2016) Bioinformatics

Deciphering the Routes of invasion of Drosophila suzukii by Means of ABC Random Forest
Fraimout et al. (2017) Molecular Biology and Evolution

ABC random forests for Bayesian parameter inference
Raynal, Marin et al. (2018) Bioinformatics

Learn on a huge reference table using Random Forests (RF)

Introduction

Reliable ABC model choice via random forests
Pudlo, Marin et al. (2016) Bioinformatics

Deciphering the Routes of invasion of Drosophila suzukii by Means of ABC Random Forest
Fraimout et al. (2017) Molecular Biology and Evolution

ABC random forests for Bayesian parameter inference
Raynal, Marin et al. (2018) Bioinformatics

Learn on a huge reference table using Random Forests (RF)

Introduction

Reliable ABC model choice via random forests
Pudlo, Marin et al. (2016) Bioinformatics

Deciphering the Routes of invasion of Drosophila suzukii by Means of ABC Random Forest
Fraimout et al. (2017) Molecular Biology and Evolution

ABC random forests for Bayesian parameter inference
Raynal, Marin et al. (2018) Bioinformatics

Learn on a huge reference table using Random Forests (RF)

Introduction

No tuning parameter, very good properties for sparse problems and heterogeneous predictors (the summary statistics)

RF have theoretical guarantees for sparse problems
Biau (2012) JMLR
Scornet, Biau, Vert (2015) The Annals of Statistics

Sub-optimal to construct RF able to estimate everywhere in the space of predictor variables

We are only interested in one point, the observed dataset $\mathbf{y}^*$

$\Rightarrow$ **local approaches**

Introduction

No tuning parameter, very good properties for sparse problems and heterogeneous predictors (the summary statistics)

RF have theoretical guarantees for sparse problems
Biau (2012) JMLR
Scornet, Biau, Vert (2015) The Annals of Statistics

Sub-optimal to construct RF able to estimate everywhere in the space of predictor variables

We are only interested in one point, the observed dataset $\mathbf{y}^*$

$\Rightarrow$ **local approaches**

Introduction

Here, we focus on model choice problems

We observe $\mathbf{y}^*$ and consider K statistical models $\mathcal{M}_1, \ldots, \mathcal{M}_K$ in competition

Prior distributions $\pi(\mathcal{M} = \mathcal{M}_k)$ and $\pi_k(\theta_k)$ on the parameters of model $\mathcal{M}_k$

Introduction

Here, we focus on model choice problems

We observe $\mathbf{y}^*$ and consider K statistical models $\mathcal{M}_1, \ldots, \mathcal{M}_K$ in competition

Prior distributions $\pi(\mathcal{M} = \mathcal{M}_k)$ and $\pi_k(\theta_k)$ on the parameters of model $\mathcal{M}_k$

Introduction

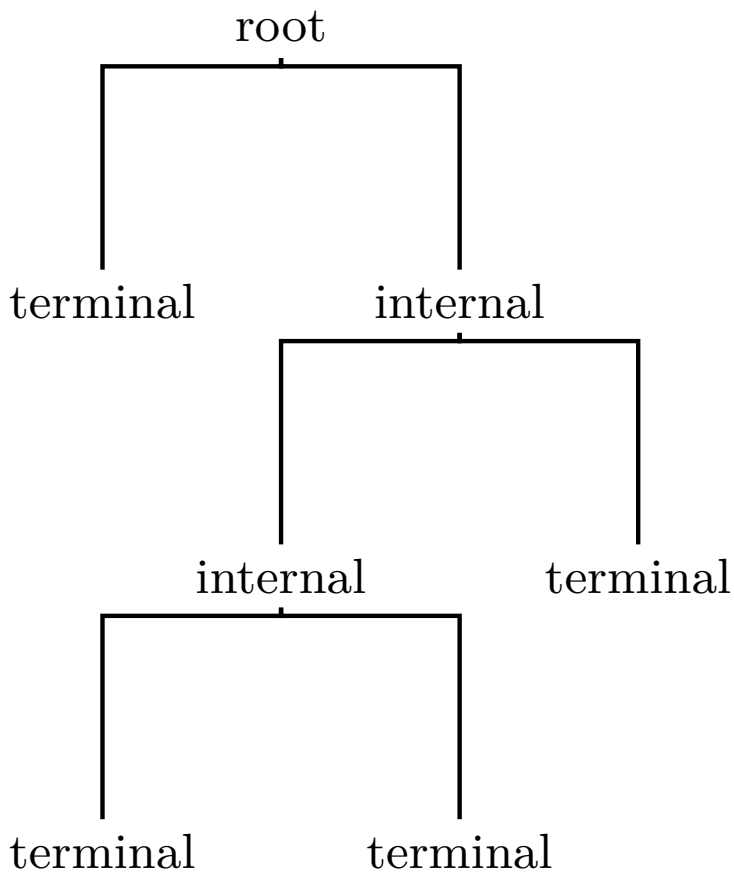
Here, we focus on model choice problems

We observe $\mathbf{y}^*$ and consider K statistical models $\mathcal{M}_1, \ldots, \mathcal{M}_K$ in competition

Prior distributions $\pi(\mathcal{M} = \mathcal{M}_k)$ and $\pi_k(\theta_k)$ on the parameters of model $\mathcal{M}_k$

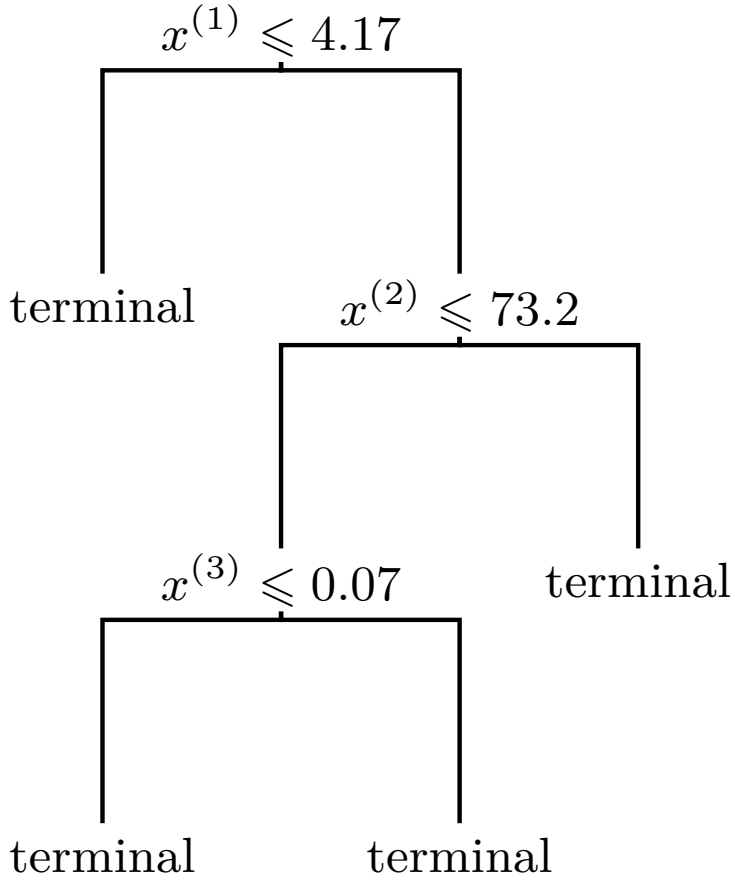
Classification trees

1. Binary trees
2. Each internal node splits the training set into two daughter nodes depending on a condition $x^{(j)} \leqslant s$
3. Predictor j and split value s chosen to maximise an information gain
4. Each terminal node (leaf) predicts a model, the prediction is the majority vote in the leaf where it ends once passed through the tree



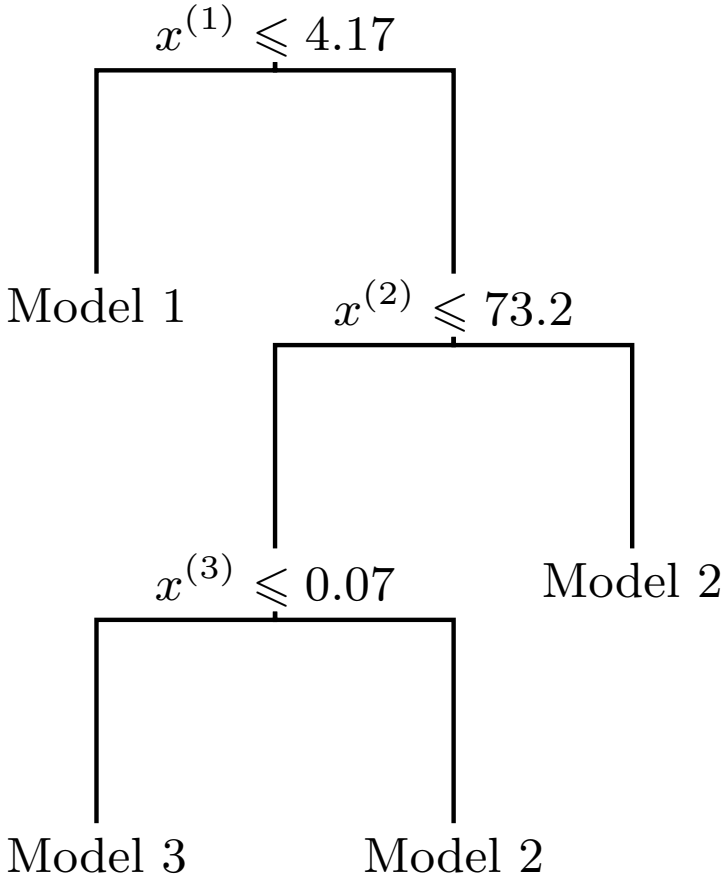
Classification trees

1. Binary trees
2. Each internal node splits the training set into two daughter nodes depending on a condition $x^{(j)} \leqslant s$
3. Predictor j and split value s chosen to maximise an information gain
4. Each terminal node (leaf) predicts a model, the prediction is the majority vote in the leaf where it ends once passed through the tree



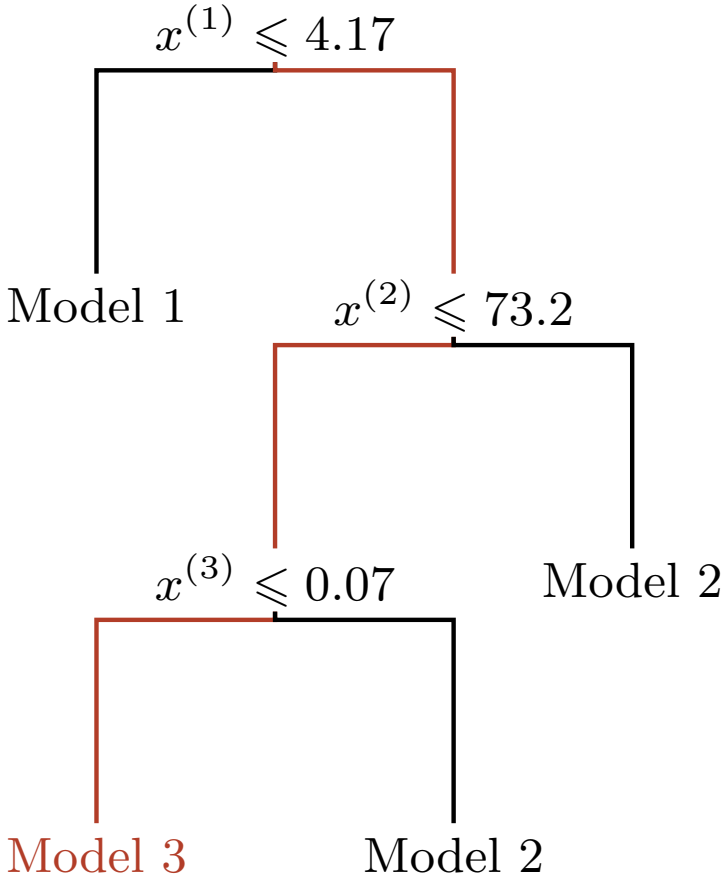
Classification trees

1. Binary trees
2. Each internal node splits the training set into two daughter nodes depending on a condition $x^{(j)} \leqslant s$
3. Predictor j and split value s chosen to maximise an information gain
4. Each terminal node (leaf) predicts a model, the prediction is the majority vote in the leaf where it ends once passed through the tree



Classification trees

1. Binary trees
2. Each internal node splits the training set into two daughter nodes depending on a condition $x^{(j)} \leqslant s$
3. Predictor j and split value s chosen to maximise an information gain
4. Each terminal node (leaf) predicts a model, the prediction is the majority vote in the leaf where it ends once passed through the tree



Classification trees

Information gain

$$\sum_{k=1}^{K} p_k(1-p_k) - \left\{ \frac{N_L}{N_{root}} \sum_{k=1}^{K} p_{k,L}(1-p_{k,L}) + \frac{N_R}{N_{root}} \sum_{k=1}^{K} p_{k,R}(1-p_{k,R}) \right\}$$

p_k corresponds to the proportion of points in the root node associated to the model k
$p_{k,L}$ the proportion for the left daughter node
$p_{k,R}$ the proportion for the right daughter node

Maximize information gain using the Gini impurity $\sum_{k=1}^{K} p_k(1-p_k)$ *or the Entropy* $-\sum_{k=1}^{K} p_k \log(p_k)$, *that's pretty much the same*

Random Forests

Grow a forest of many trees

Grow each tree on an independent bootstrap sample from the training data such that at each node:

1. Select m variables at random out of all predictors
2. Find the best split on the selected m predictors

Vote the trees to get predictions for new data

Improve on CART with respect to accuracy and stability

Random Forests

Grow a forest of many trees

Grow each tree on an independent bootstrap sample from the training data such that at each node:

1. Select m variables at random out of all predictors
2. Find the best split on the selected m predictors

Vote the trees to get predictions for new data

Improve on CART with respect to accuracy and stability

Random Forests

Grow a forest of many trees

Grow each tree on an independent bootstrap sample from the training data such that at each node:

1. Select m variables at random out of all predictors
2. Find the best split on the selected m predictors

Vote the trees to get predictions for new data

Improve on CART with respect to accuracy and stability

Random Forests

Grow a forest of many trees

Grow each tree on an independent bootstrap sample from the training data such that at each node:

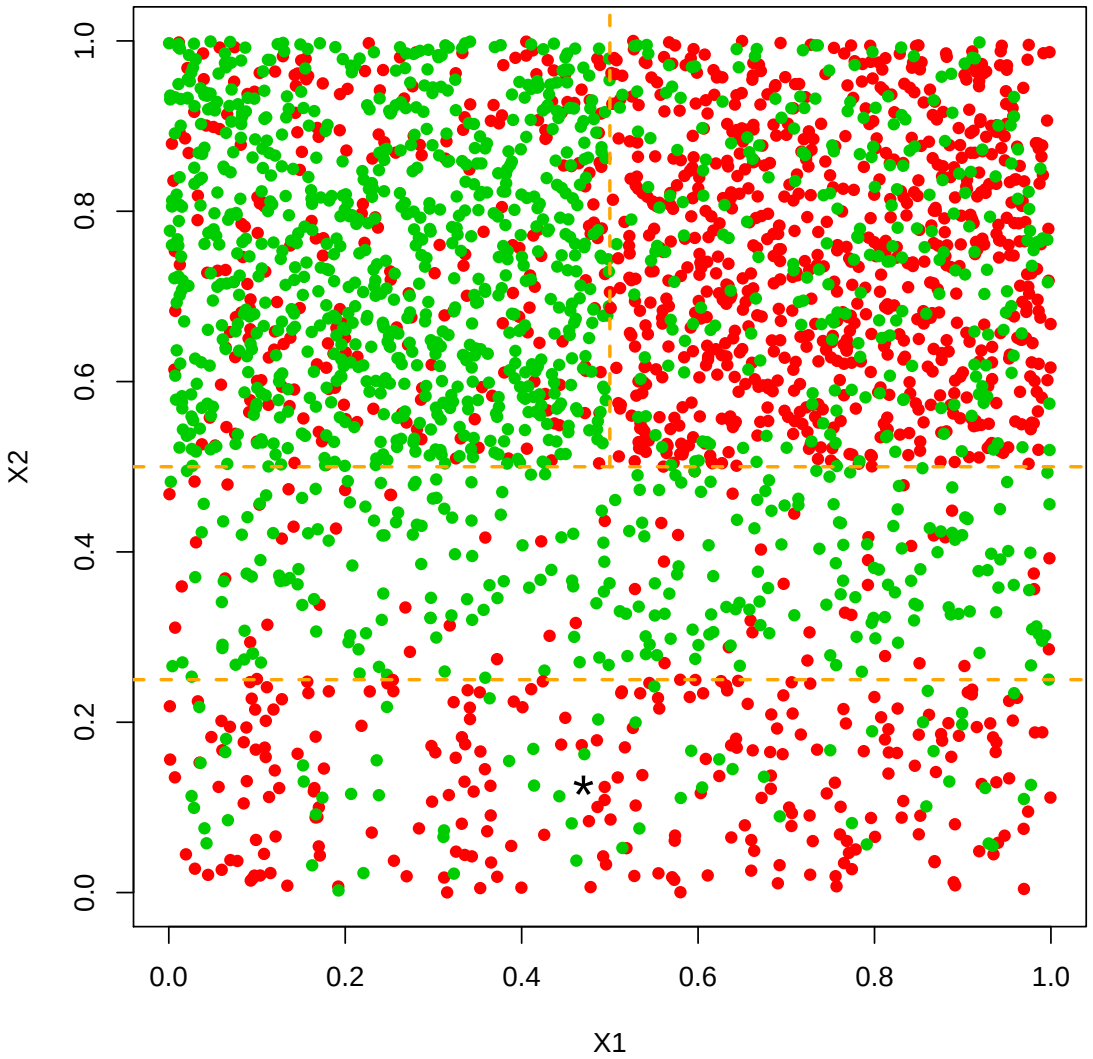
1. Select m variables at random out of all predictors
2. Find the best split on the selected m predictors

Vote the trees to get predictions for new data

Improve on CART with respect to accuracy and stability

Data fragmentation

Data fragmentation problem



A standard decision tree will split at $x^{(1)} \approx 0.5$
$\Rightarrow$ loosing some interesting datasets for the prediction of $\mathbf{y}^*$
We would like to split according to the pertinent predictors for $\mathbf{y}^*$

ABC Random Forests

Input ABC reference table involving model index and summary statistics, table used as learning set

possibly large collection of summary statistics: from scientific theory input to machine-learning alternatives

For $i = 1, \ldots, N$

a) Generate m_i from the prior $\pi(\mathcal{M} = m)$

b) Generate $\theta^i_{m_i}$ from the prior $\pi_{m_i}(\cdot)$

c) Generate $\mathbf{z}_i$ from the model $f_{m_i}(\cdot|\theta^i_{m_i})$

d) Calculate $\mathbf{x}_i = \eta(\mathbf{z}_i)$

Output a random forest classifier to infer model indexes $\widehat{m(\eta(\mathbf{y}^*))}$

ABC Random Forests

Input ABC reference table involving model index and summary statistics, table used as learning set

possibly large collection of summary statistics: from scientific theory input to machine-learning alternatives

For $i = 1, \ldots, N$

a) Generate m_i from the prior $\pi(\mathcal{M} = m)$

b) Generate $\theta^i_{m_i}$ from the prior $\pi_{m_i}(\cdot)$

c) Generate $\mathbf{z}_i$ from the model $f_{m_i}(\cdot|\theta^i_{m_i})$

d) Calculate $\mathbf{x}_i = \eta(\mathbf{z}_i)$

Output a random forest classifier to infer model indexes $\widehat{m(\eta(\mathbf{y}^*))}$

ABC Random Forests

Input ABC reference table involving model index and summary statistics, table used as learning set

possibly large collection of summary statistics: from scientific theory input to machine-learning alternatives

For $i = 1, \ldots, N$

a) Generate m_i from the prior $\pi(\mathcal{M} = m)$

b) Generate $\boldsymbol{\theta}'_{m_i}$ from the prior $\pi_{m_i}(\cdot)$

c) Generate $\mathbf{z}_i$ from the model $f_{m_i}(\cdot|\boldsymbol{\theta}'_{m_i})$

d) Calculate $\mathbf{x}_i = \eta(\mathbf{z}_i)$

Output a random forest classifier to infer model indexes $\widehat{m(\eta(\mathbf{y}^*))}$

ABC Random Forests

Input ABC reference table involving model index and summary statistics, table used as learning set

possibly large collection of summary statistics: from scientific theory input to machine-learning alternatives

For $i = 1, \ldots, N$

a) Generate m_i from the prior $\pi(\mathcal{M} = m)$

b) Generate $\boldsymbol{\theta}'_{m_i}$ from the prior $\pi_{m_i}(\cdot)$

c) Generate $\mathbf{z}_i$ from the model $f_{m_i}(\cdot|\boldsymbol{\theta}'_{m_i})$

d) Calculate $\mathbf{x}_i = \eta(\mathbf{z}_i)$

Output a random forest classifier to infer model indexes $\widehat{m(\eta(\mathbf{y}^*))}$

Local splitting rules

Change the information gain to the benefit of a more local one

Use $\mathbf{y}^*$ to drive the splits and thus the tree construction

Uni-dimensional Kernel approach

Local information gain associated to variable j

$$\sum_{k=1}^{K} \tilde{p}_k(1-\tilde{p}_k) - \left\{ \frac{\tilde{N}_L}{\tilde{N}_{root}} \sum_{k=1}^{K} (\tilde{p}_{k,L}(1-\tilde{p}_{k,L}) + \frac{\tilde{N}_R}{\tilde{N}_{root}} \sum_{k=1}^{K} (\tilde{p}_{k,R}(1-\tilde{p}_{k,R}) \right\}$$

$$\tilde{p}_{k,L} = \sum_{i \in \text{LD}} K_{h_j} \left(\eta(\mathbf{y}^*)^{(j)} - \mathbf{x}_i^{(j)} \right) \mathbb{I}_{m_i=k}$$

$$\tilde{N}_L = \sum_{i \in \text{ROOT}} K_{h_j} \left(\eta(\mathbf{y}^*)^{(j)} - \mathbf{x}_i^{(j)} \right) \mathbb{I}_{\mathbf{x}_i^{(j)} \leq s}$$

Local splitting rules

We tried several kernels and bandwidths

That is related to

Lazy Decision Trees
Friedman et al. (1997) Proc. of the 13th National Conference on AAAI

Local weighting of the individuals

Case-Specific Random Forests
Xu et al. (2016) Journal of Computational and Graphical Statistics

Produce a first RF with default parameter

Deduce from that RF distances between individuals in the training set and y^*

Produce a second RF that uses these weights to modify the bootstrap step

We tried another possibility: identify neighbours using standard distance and then construct a RF on this set of neighbours

Local weighting of the individuals

Case-Specific Random Forests
Xu et al. (2016) Journal of Computational and Graphical Statistics

Produce a first RF with default parameter

Deduce from that RF distances between individuals in the training set and y^*

Produce a second RF that uses these weights to modify the bootstrap step

We tried another possibility: identify neighbours using standard distance and then construct a RF on this set of neighbours

Local weighting of the individuals

Case-Specific Random Forests
Xu et al. (2016) Journal of Computational and Graphical Statistics

Produce a first RF with default parameter

Deduce from that RF distances between individuals in the training set and y^*

Produce a second RF that uses these weights to modify the bootstrap step

We tried another possibility: identify neighbours using standard distance and then construct a RF on this set of neighbours

Local weighting of the individuals

Case-Specific Random Forests
Xu et al. (2016) Journal of Computational and Graphical Statistics

Produce a first RF with default parameter

Deduce from that RF distances between individuals in the training set and y^*

Produce a second RF that uses these weights to modify the bootstrap step

We tried another possibility: identify neighbours using standard distance and then construct a RF on this set of neighbours

Local weighting of the predictors

Produce a first RF with default parameter

Deduce from that RF the predictors that are important to predict $\mathbf{y}^*$

1. pass $\mathbf{y}^*$ through each tree of the RF and count the number of times each has been used in a splitting rule to allocate $\mathbf{y}^*$
2. deduce predictors weights

Produce a second RF that uses these weights to randomly select the predictors at each node

Weight the predictors: better cuts but increase the correlation between the RF's trees...

Local weighting of the predictors

Produce a first RF with default parameter

Deduce from that RF the predictors that are important to predict $\mathbf{y}^*$

1. pass $\mathbf{y}^*$ through each tree of the RF and count the number of times each has been used in a splitting rule to allocate $\mathbf{y}^*$

2. deduce predictors weights

Produce a second RF that uses these weights to randomly select the predictors at each node

Weight the predictors: better cuts but increase the correlation between the RF's trees...

Local weighting of the predictors

Produce a first RF with default parameter

Deduce from that RF the predictors that are important to predict $\mathbf{y}^*$

1. pass $\mathbf{y}^*$ through each tree of the RF and count the number of times each has been used in a splitting rule to allocate $\mathbf{y}^*$

2. deduce predictors weights

Produce a second RF that uses these weights to randomly select the predictors at each node

Weight the predictors: better cuts but increase the correlation between the RF's trees...

Local weighting of the predictors

Produce a first RF with default parameter

Deduce from that RF the predictors that are important to predict $\mathbf{y}^*$

1. pass $\mathbf{y}^*$ through each tree of the RF and count the number of times each has been used in a splitting rule to allocate $\mathbf{y}^*$

2. deduce predictors weights

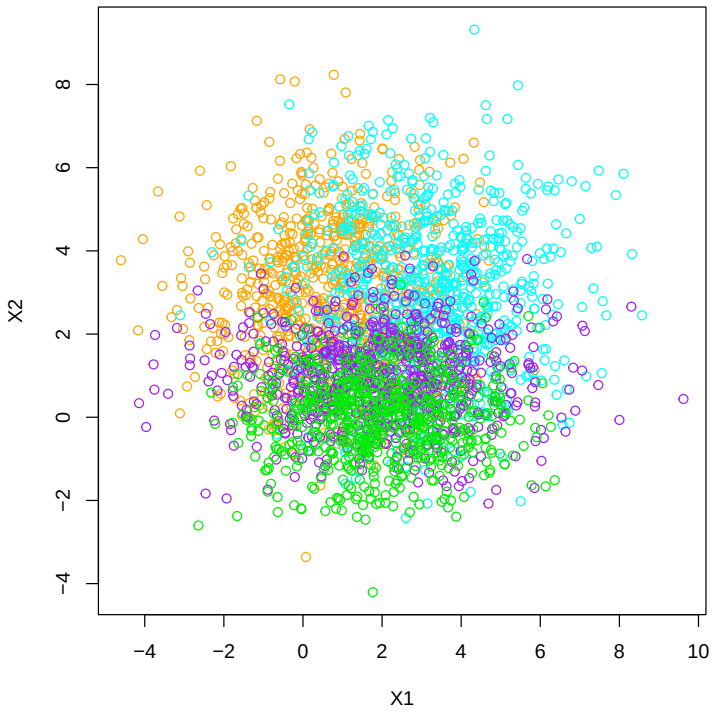
Produce a second RF that uses these weights to randomly select the predictors at each node

Weight the predictors: better cuts but increase the correlation between the RF's trees...

Numerical results

We tried several examples, for instance, a 20-dimensional Gaussian mixtures with 4 classes and 20 noise predictor variables

Simple enough to compute the Bayes classifier



Numerical results

$3,000$ instances in the training set, sampled among the four classes with equal probabilities
500 instances as testing set

100 trees in the forests

1. Bayes classifier: 12.6
2. RF: 22.4
3. Local splitting rule: same as RF after tuning
4. Local weighting of the individuals: same as RF after tuning
5. Local weighting of the predictors: same as RF

Numerical results

$3,000$ instances in the training set, sampled among the four classes with equal probabilities
500 instances as testing set

100 trees in the forests

1. Bayes classifier: 12.6
2. RF: 22.4
3. Local splitting rule: same as RF after tuning
4. Local weighting of the individuals: same as RF after tuning
5. Local weighting of the predictors: same as RF

Numerical results

$3,000$ instances in the training set, sampled among the four classes with equal probabilities
500 instances as testing set

100 trees in the forests

1. Bayes classifier: 12.6
2. RF: 22.4
3. Local splitting rule: same as RF after tuning
4. Local weighting of the individuals: same as RF after tuning
5. Local weighting of the predictors: same as RF

Discussion

Local forest methods do not work as expected

Last year, I was invited to give a talk at a NIPS workshop on PAC-Bayesian methods, I discussed with people about **local, transductive, case-specific, lazy, instance-based**... learning strategies

In a Chinese restaurant, I got a hidden message within a cake *This year, take comfort in your rituals, but be open to new experiences*
I was very enthusiastic

One year latter, back to the real world.... trees methods are extremely difficult to localize

Thank you very much for your attention

Discussion

Local forest methods do not work as expected

Last year, I was invited to give a talk at a NIPS workshop on PAC-Bayesian methods, I discussed with people about **local, transductive, case-specific, lazy, instance-based**... learning strategies

In a Chinese restaurant, I got a hidden message within a cake *This year, take comfort in your rituals, but be open to new experiences*
I was very enthusiastic

One year latter, back to the real world.... trees methods are extremely difficult to localize

Thank you very much for your attention

Discussion

Local forest methods do not work as expected

Last year, I was invited to give a talk at a NIPS workshop on PAC-Bayesian methods, I discussed with people about **local, transductive, case-specific, lazy, instance-based**... learning strategies

In a Chinese restaurant, I got a hidden message within a cake *This year, take comfort in your rituals, but be open to new experiences*
I was very enthusiastic

One year latter, back to the real world.... trees methods are extremely difficult to localize

Thank you very much for your attention

Discussion

Local forest methods do not work as expected

Last year, I was invited to give a talk at a NIPS workshop on PAC-Bayesian methods, I discussed with people about **local, transductive, case-specific, lazy, instance-based**... learning strategies

In a Chinese restaurant, I got a hidden message within a cake *This year, take comfort in your rituals, but be open to new experiences*
I was very enthusiastic

One year latter, back to the real world.... trees methods are extremely difficult to localize

Thank you very much for your attention

Discussion

Local forest methods do not work as expected

Last year, I was invited to give a talk at a NIPS workshop on PAC-Bayesian methods, I discussed with people about **local, transductive, case-specific, lazy, instance-based**... learning strategies

In a Chinese restaurant, I got a hidden message within a cake *This year, take comfort in your rituals, but be open to new experiences*
I was very enthusiastic

One year latter, back to the real world.... trees methods are extremely difficult to localize

Thank you very much for your attention