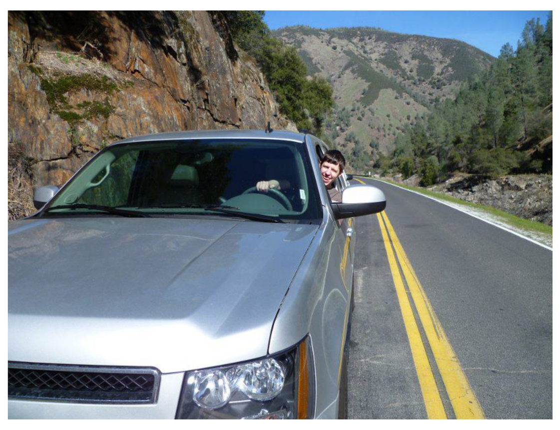
Random forests with random projections of the output space for high dimensional multi-label classification

Arnaud Joly, Pierre Geurts, Louis Wehenkel
a.joly@ulg.ac.be @JolyArnaud

Multilabel classification

Given a set of n samples of input-output pairs $((x^i, y^i) \in (\mathcal{X} \times \mathcal{Y}))_{i=1}^n$, a supervised learning task is defined as searching for the function $f : \mathcal{X} \to \mathcal{Y}$ in a hypothesis space that minimizes some loss function over the joint distribution of input-output pairs.
In multi-label classification, y^i is a subset of the label space $\mathcal{Y}$ of size p.

Input $\mathcal{X}$ 800×600 pixel



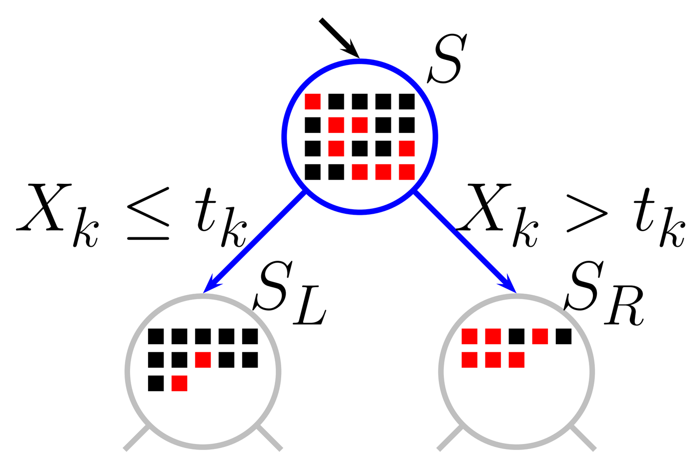
Output $\mathcal{Y}$ labels

This image can be labelled with "car","person", "mountain", but not with "house" or "elephant".

If each label corresponds to a wikipedia article, then we have around 4 million labels.

Random forest

Randomized trees are built on a bootstrap copy of the samples by recursively maximizing the reduction of impurity, here the variance Var. At each node, the best split is selected among k randomly selected features.



$$\mathrm{Var}(S) = 0.24$$
$$\mathrm{Var}(S_l) = 0.014$$
$$\mathrm{Var}(S_R) = 0.1875$$
$$\Delta \mathrm{Var}(S) = \mathrm{Var}(S) - \frac{12}{20}\mathrm{Var}(S_L) - \frac{8}{20}\mathrm{Var}(t_R) \approx 0.16$$

High dimensional output space $\mathcal{Y}$ is a bottleneck of random forest

Easy to have a very high number of labels. . . $\longrightarrow$ Tree growing algorithm requires the computation of the sum of the variance

- over the label space
- at each tree node and
- for each candidate split.

Solution Multi-output regression trees in randomly projected output space

Methods

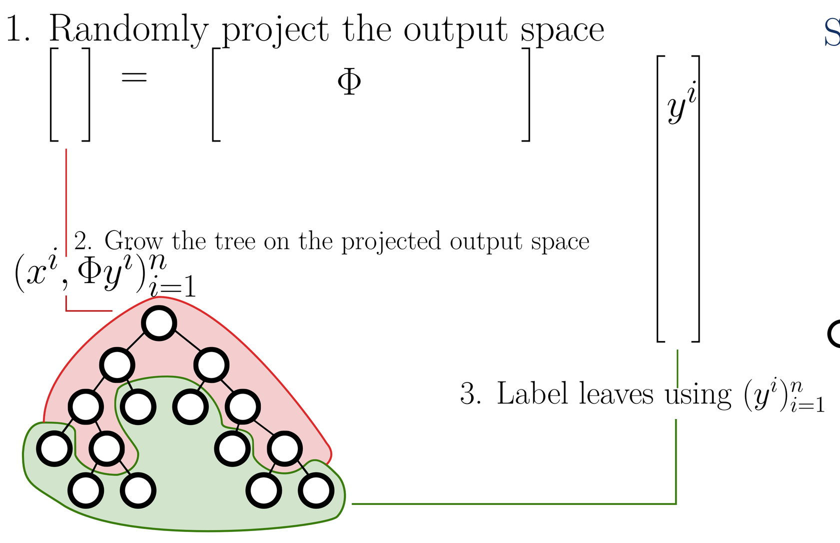
We propose to approximate the computation of the variance by using random projection of the output space.

Theorem

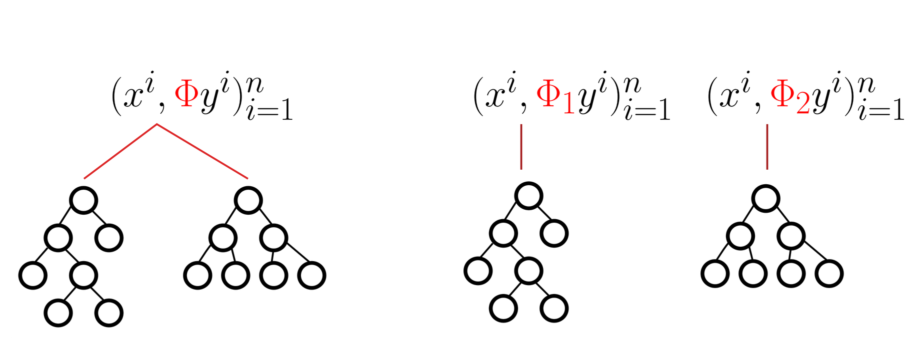
Given $\epsilon > 0$, a sample $(y^i)_{i=1}^n$ of n points $y \in \mathbb{R}^d$, and a projection matrix $\Phi \in \mathbb{R}^{m \times d}$ such that for all pairs of points the Jonhson-Lindenstrauss lemma holds, we have also

$$(1-\epsilon)\,\mathrm{Var}\left((y^i)_{i=1}^n\right) \leq \mathrm{Var}\left((\Phi y^i)_{i=1}^n\right) \leq (1+\epsilon)\,\mathrm{Var}\left((y^i)_{i=1}^n\right).$$

Single tree



Ensemble of randomized trees



Bias-variance analysis

Averaging over the learning set LS, algorithm randomization ϵ and output subspace randomization Φ, the square error Err of t multi output tree models can be decomposed into:

Single shared subspace (Algo 1)

$$E_{LS,\Phi,\varepsilon^t}\{Err(f_1(x; LS, \Phi, \varepsilon^t))\} = \sigma_R^2(x) + B^2(x) + V_{LS}(x) + \frac{V_{Algo}(x)}{t} + V_{Proj}(x).$$

Individual subspace (Algo 2)

$$E_{LS,\Phi^t,\varepsilon^t}\{Err(f_2(x; LS, \Phi^t, \varepsilon^t))\} = \underbrace{\sigma_R^2(x)}_{\text{residual error}} + \underbrace{B^2(x)}_{\text{bias}} + \underbrace{V_{LS}(x) + \frac{V_{Algo}(x) + V_{Proj}(x)}{t}}_{\text{variance}}.$$

If the additional computational burden needed to generate a different random projection for each tree is not problematic, then individual subspace should always be preferred to single shared subspace.

Conclusion

- Lower computing time, without affecting accuracy.
- Optimizing input and output randomization could improve prediction performance.

Source code is available @ `github.com/arjoly`.

Future work

Develop efficient technique to adjust random output space parameters so to reach the best accuracy and computing time trade-off.

Experiments

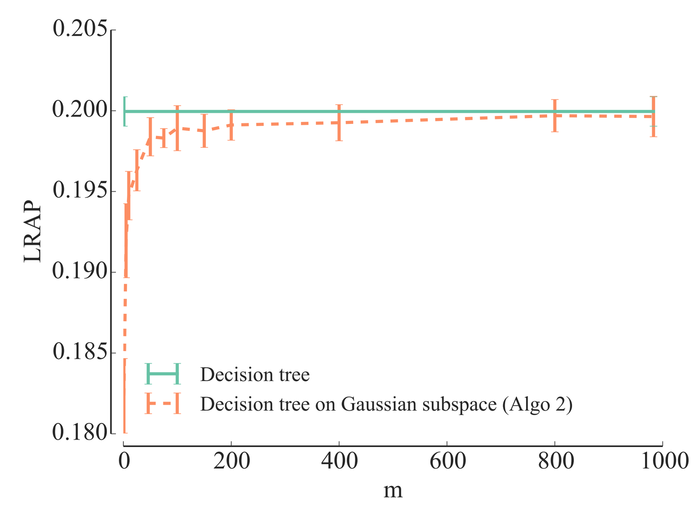
Label ranking average precision to assess performance

$$\mathrm{LRAP}(\hat{f}) = \frac{1}{|TS|}\sum_{i \in TS}\frac{1}{|y^i|}\sum_{j \in \{k : y_k^i = 1\}}\frac{|\mathcal{L}_j^i(y^i)|}{|\mathcal{L}_j^i(1_d)|}, \text{ with } \mathcal{L}_j^i(q) = \left\{k : q_k = 1 \text{ and } \hat{f}(x^i)_k \geq \hat{f}(x^i)_j\right\}$$

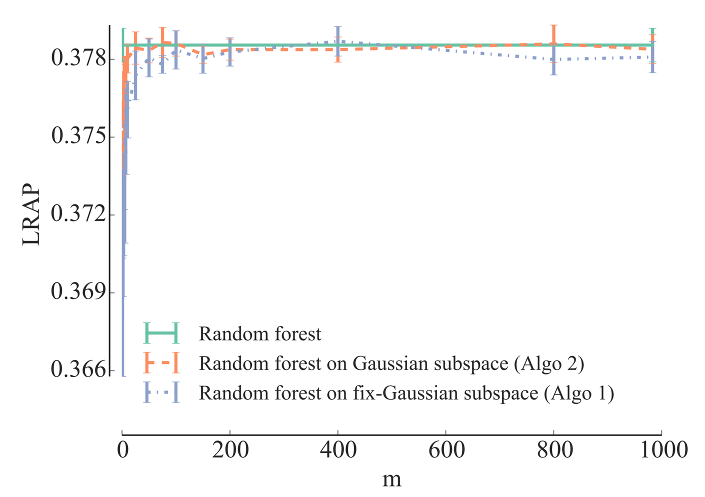
where $\hat{f}(x^i)_j$ is the probability (or the score) associated to the label j by the learnt model $\hat{f}$ applied to x^i, 1_d is a d-dimensional row vector of ones. Higher score if true labels have a higher probability (score) than the false labels.

"Delicious" dataset (983 labels, 100 trees, $k = \sqrt{p}$, no pruning)

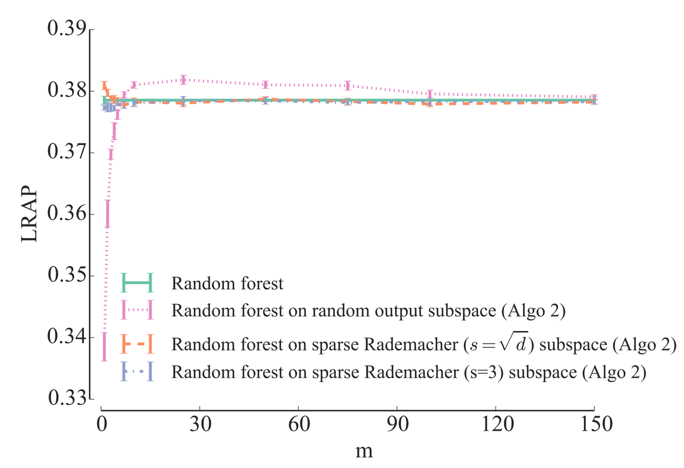
Randomly projecting the output space reduces computing time of random forest from 3458 seconds (no projection) to 311 seconds ($m = 25$, Gaussian individual subspace) without accuracy degradation.



(a) Decision tree performance converges with $m = 200$ Gaussian random output projections

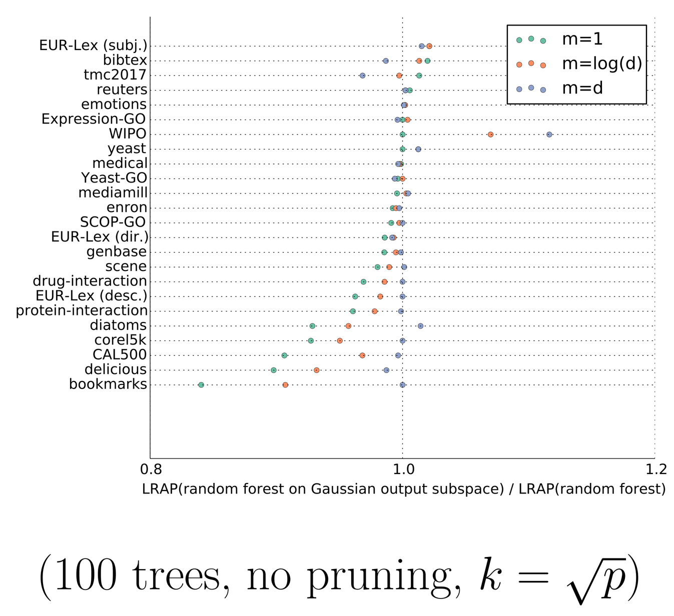


(b) Faster convergence with ensemble of randomized trees on individual output space ($m = 25$).



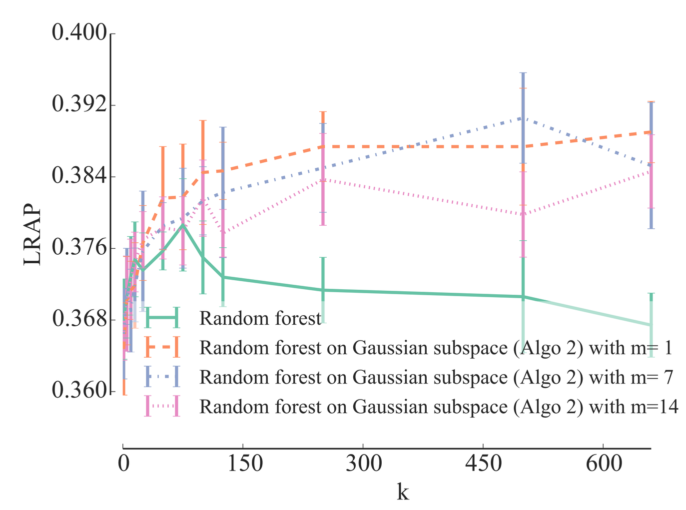
(c) Sparse random projection output sub-space yield better average precision than on the original output space.

Systematic analysis on 24 datasets



(100 trees, no pruning, $k = \sqrt{p}$)

Output randomization could be more effective than input randomization.



Drug-interaction dataset
(1554 labels, 100 trees, no pruning)