

# Random Forest

## 一、Random Forest Algorithm

1. 想到tree就会想到森林，我们试图用aggregation of aggregation来提高算法的效果。

Random Forest

Random Forest Algorithm

Random Forest (RF)

**random forest (RF) = bagging + fully-grown C&RT decision tree**

```
function RandomForest( $\mathcal{D}$ )
  For  $t = 1, 2, \dots, T$ 
    ① request size- $N'$  data  $\tilde{\mathcal{D}}_t$  by
      bootstrapping with  $\mathcal{D}$ 
    ② obtain tree  $g_t$  by DTree( $\tilde{\mathcal{D}}_t$ )
  return  $G = \text{Uniform}(\{g_t\})$ 
```

```
function DTree( $\mathcal{D}$ )
  if termination return base  $g_t$ 
  else
    ① learn  $b(\mathbf{x})$  and split  $\mathcal{D}$  to
       $\mathcal{D}_c$  by  $b(\mathbf{x})$ 
    ② build  $G_c \leftarrow \text{DTree}(\mathcal{D}_c)$ 
    ③ return  $G(\mathbf{x}) =$ 
      
$$\sum_{c=1}^C \mathbb{I}[b(\mathbf{x}) = c] G_c(\mathbf{x})$$

```

- highly **parallel/efficient** to learn
- **inherit pros** of C&RT
- **eliminate cons** of fully-grown tree

对比一下，由于数据量有限，我们使用bootstrap来随机取得到 $N'$ 大小的样本然后学习出一颗decision tree然后再对每一棵树做uniform vote，bagging然后就得到了最后的G

注意我们这里用的是 **fully grown**的tree而不是通过剪枝得到的树，实际上加入剪枝操作会效果更好，这里我们就考虑randomness这一点上我们可以改善多个树的作用

2. 我们此处使用的RF算法实际上是取了样本的一个随机的子空间来形成树所以总结出RF的特点如下：

$$\text{RF} = \text{bagging} + \text{random-subspace C\&RT}$$

## 二、Out-Of-Bag (OOB) Estimate

1. 首先回顾一下bagging算法：

# Bagging Revisited

## Bagging

function  $\text{Bag}(\mathcal{D}, \mathcal{A})$

For  $t = 1, 2, \dots, T$

① request size- $N'$  data  $\tilde{\mathcal{D}}_t$   
by **bootstrapping** with  $\mathcal{D}$

② obtain base  $g_t$  by  $\mathcal{A}(\tilde{\mathcal{D}}_t)$

return  $G = \text{Uniform}(\{g_t\})$

	$g_1$	$g_2$	$g_3$	$\dots$	$g_T$
$(\mathbf{x}_1, y_1)$	$\tilde{\mathcal{D}}_1$	*	$\tilde{\mathcal{D}}_3$		$\tilde{\mathcal{D}}_T$
$(\mathbf{x}_2, y_2)$	*	*	$\tilde{\mathcal{D}}_3$		$\tilde{\mathcal{D}}_T$
$(\mathbf{x}_3, y_3)$	*	$\tilde{\mathcal{D}}_2$	*		$\tilde{\mathcal{D}}_T$
$\dots$					
$(\mathbf{x}_N, y_N)$	$\tilde{\mathcal{D}}_1$	$\tilde{\mathcal{D}}_2$	*		*

\* in  $t$ -th column: not used for obtaining  $g_t$   
—called **out-of-bag (OOB) examples** of  $g_t$

2. 我们通过随机取样的方法来获得  $\tilde{\mathcal{D}}_t$ ，现在我们考虑没有被选择中的样本，在这里我们称之为 **OOB examples**，我们对于  $N' = N$  情况下 OOB 的样本概率进行估计，

$$(1 - \frac{1}{N})^N \approx \frac{1}{e} \quad (1)$$

3. 我们对于 OOB 和 Validation 做一个比较:

## OOB versus Validation

### OOB

	$g_1$	$g_2$	$g_3$	$\dots$	$g_T$
$(\mathbf{x}_1, y_1)$	$\tilde{\mathcal{D}}_1$	*	$\tilde{\mathcal{D}}_3$		$\tilde{\mathcal{D}}_T$
$(\mathbf{x}_2, y_2)$	*	*	$\tilde{\mathcal{D}}_3$		$\tilde{\mathcal{D}}_T$
$(\mathbf{x}_3, y_3)$	*	$\tilde{\mathcal{D}}_2$	*		$\tilde{\mathcal{D}}_T$
$\dots$					
$(\mathbf{x}_N, y_N)$	$\tilde{\mathcal{D}}_1$	*	*		*

### Validation

	$g_1^-$	$g_2^-$	$\dots$	$g_M^-$
$\mathcal{D}_{\text{train}}$	$\mathcal{D}_{\text{train}}$			$\mathcal{D}_{\text{train}}$
$\mathcal{D}_{\text{val}}$	$\mathcal{D}_{\text{val}}$			$\mathcal{D}_{\text{val}}$
$\mathcal{D}_{\text{val}}$	$\mathcal{D}_{\text{val}}$			$\mathcal{D}_{\text{val}}$
$\mathcal{D}_{\text{train}}$	$\mathcal{D}_{\text{train}}$			$\mathcal{D}_{\text{train}}$

- \* like  $\mathcal{D}_{\text{val}}$ : 'enough' random examples unused during training
- use \* to validate  $g_t$ ? easy, but **rarely needed**
- use \* to validate  $G$ ?  $E_{\text{OOB}}(G) = \frac{1}{N} \sum_{n=1}^N \text{err}(y_n, G_n^-(\mathbf{x}_n))$ ,  
with  $G_n^-$  contains only trees that  $\mathbf{x}_n$  is OOB of,  
such as  $G_N^-(\mathbf{x}) = \text{average}(g_2, g_3, g_T)$

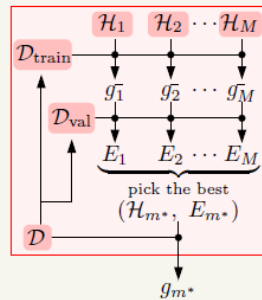
$E_{\text{OOB}}$ : **self-validation** of bagging/RF

4. 我们可以通过衡量  $E_{\text{OOB}}$  来衡量 RF 的效果，这种可以看作是 RF 的一个 **self-validation** 的优势！  
因此我们也可以利用  $E_{\text{OOB}}$  来选择模型，一般情况下它甚至比一般的 validation 效果要好

## Model Selection by OOB Error

### Previously: by Best $E_{\text{val}}$

$$\begin{aligned} g_{m^*} &= \mathcal{A}_{m^*}(\mathcal{D}) \\ m^* &= \underset{1 \leq m \leq M}{\operatorname{argmin}} E_m \\ E_m &= E_{\text{val}}(\mathcal{A}_m(\mathcal{D}_{\text{train}})) \end{aligned}$$



### RF: by Best $E_{\text{oob}}$

$$\begin{aligned} G_{m^*} &= \text{RF}_{m^*}(\mathcal{D}) \\ m^* &= \underset{1 \leq m \leq M}{\operatorname{argmin}} E_m \\ E_m &= E_{\text{oob}}(\text{RF}_m(\mathcal{D})) \end{aligned}$$

- use  $E_{\text{oob}}$  for **self-validation** —of RF **parameters** such as  $d''$
- **no re-training** needed

$E_{\text{oob}}$  often **accurate** in practice

## 三、Feature Selection

1. decision tree是一个具有内在特征选择机制的模型，这在大部分模型当中是很少出现的：

### Feature Selection

for  $\mathbf{x} = (x_1, x_2, \dots, x_d)$ , want to remove

- **redundant** features: like keeping one of 'age' and 'full birthday'
- **irrelevant** features: like insurance type for cancer prediction

and only 'learn' **subset-transform**  $\Phi(\mathbf{x}) = (x_{i_1}, x_{i_2}, x_{i_{d'}})$   
with  $d' < d$  for  $g(\Phi(\mathbf{x}))$

advantages:

- **efficiency**: simpler hypothesis and shorter prediction time
- **generalization**: 'feature noise' removed
- **interpretability**

disadvantages:

- **computation**: 'combinatorial' optimization in training
- **overfit**: 'combinatorial' selection
- **mis-interpretability**

decision tree: a rare model  
with **built-in feature selection**

可以达到摒弃不相关和多余特征的效果，类似于一种降维的方法

2. 总结一下决策树的优劣：

- 优势：
  - 高效：用简单的假设构成的预测
  - 泛化能力：摒弃了特征的噪声（不相关和多余的特征）
  - 可解释性：符合人类决策
- 劣势：

- 计算力：训练过程时间长
- 过拟合：特征选择过多
- 解释性差：理论基础不坚固

3. **特征选择机制：重要性原则，适用于线性模型**，权重的大小代表着重要性

### Feature Selection by Importance

idea: if possible to calculate

$\text{importance}(i)$  for  $i = 1, 2, \dots, d$

then can select  $i_1, i_2, \dots, i_{d'}$  of top- $d'$  importance

#### importance by linear model

$$\text{score} = \mathbf{w}^T \mathbf{x} = \sum_{i=1}^d w_i x_i$$

- intuitive estimate:  $\text{importance}(i) = |w_i|$  with some 'good'  $\mathbf{w}$
- getting 'good'  $\mathbf{w}$ : learned from data
- non-linear models? often **much harder**

4. **特征选择机制：全排列测试，适用于非线性模型**，去掉此特征后，表现的衰弱程度作为评判重要性的原则！

### Feature Importance by Permutation Test

idea: random test

—if feature  $i$  needed, 'random' values of  $x_{n,i}$  degrades performance

- which random values?
  - uniform, Gaussian, ...:  $P(x_i)$  changed
  - bootstrap, **permutation** (of  $\{x_{n,i}\}_{n=1}^N$ ):  $P(x_i)$  approximately remained
- **permutation** test:

$$\text{importance}(i) = \text{performance}(\mathcal{D}) - \text{performance}(\mathcal{D}^{(p)})$$

with  $\mathcal{D}^{(p)}$  is  $\mathcal{D}$  with  $\{x_{n,i}\}$  replaced by **permuted**  $\{x_{n,i}\}_{n=1}^N$

**permutation** test: a general statistical tool for arbitrary non-linear models like RF

5. 在原始的随机森林模型中我们采用的特征重要性度量方法：

我们结合permutation作为重要性度量，利用OOB-Validation作为Error度量

## 四、Random Forest in Action

需要足够多的树来保证稳定性！