Lecture 10: Random Forest

Hsuan-Tien Lin (林軒田)
htlin@csie.ntu.edu.tw

Department of Computer Science
& Information Engineering

National Taiwan University
(國立台灣大學資訊工程系)
Lecture 10: Random Forest

- Basic Random Forest Algorithm
- Out-Of-Bag Estimate
- Feature Selection
Random Forest

Basic Random Forest Algorithm

Recall: Bagging and Decision Tree

Bagging

function Bag(\(\mathcal{D}, \mathcal{A}\))
For \(t = 1, 2, \ldots, T\)

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

—reduces variance by voting/averaging

Decision Tree

function DT\(\text{Tree}(\mathcal{D})\)
if termination return base \(g_t\)
else

1. learn \(b(x)\) and split \(\mathcal{D}\) to \(\mathcal{D}_c\) by \(b(x)\)
2. build \(G_c \leftarrow \text{DT\text{ree}(\mathcal{D}_c)}\)
3. return \(G(x) = \sum_{c=1}^{C} [b(x) = c] G_c(x)\)

—large variance especially if fully-grown

putting them together?
(i.e. aggregate two aggregation models : - ) )
Random Forest

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

function RandomForest(\(\mathcal{D}\))
For \(t = 1, 2, \ldots, T\)
1. request size-\(N\) data \(\tilde{\mathcal{D}}_t\) by bootstrapping with \(\mathcal{D}\)
2. obtain base \(G_t\) 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 \(b(x)\) and split \(\mathcal{D}\) to \(\mathcal{D}_c\) by \(b(x)\)
2. build \(G_c \leftarrow \text{DTree}(\mathcal{D}_c)\)
3. return \(G(x) = \sum_{c=1}^{C} [b(x) = c] G_c(x)\)

• highly parallel/efficient to learn
• inherit pros of C&RT
• eliminate cons of fully-grown tree
Diversifying by Feature Projection

recall: **data randomness** for **diversity** in bagging

randomly sample $N$ examples from $\mathcal{D}$

other possibility:

randomly sample $d'$ features from $\mathbf{x}$

- chosen index $i_1, i_2, \ldots, i_{d'}$
  
  $\Phi(\mathbf{x}) = (x_{i_1}, x_{i_2}, \ldots, x_{i_{d'}})$

- $\mathcal{Z} \in \mathbb{R}^{d'}$: a **random subspace** of $\mathcal{X} \in \mathbb{R}^d$

- often $d' \ll d$, efficient when $d$ large
  
  —can be generally used for other learning models

- original RF re-sample new subspace for each $b(\mathbf{x})$ in C&RT

RF = bagging + random-subspace C&RT
Diversifying by Feature Expansion

randomly sample $d'$ features from $\mathbf{x}$: $\Phi(\mathbf{x}) = \mathbf{P} \cdot \mathbf{x}$ with row $i$ of $\mathbf{P}$ randomly $\in$ natural basis

more powerful features: row $i$ of $\mathbf{P}$ other than natural basis

- low-dimensional random projection (combination) with $\mathbf{u}$:

$$\phi_i(\mathbf{x}) = \sum_{j=1}^{d''} u_j x_j$$

- includes random subspace as a special case: $d'' = 1$ and $u_1 = 1$
- original RF consider $d'$ random projections for each $b(\mathbf{x})$ in C&RT

RF = bagging + random-combination C&RT
—randomness everywhere!
Fun Time
Bagging Revisited

Bagging

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

For \( t = 1, 2, \ldots, T \)

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

<table>
<thead>
<tr>
<th></th>
<th>( g_1 )</th>
<th>( g_2 )</th>
<th>( g_3 )</th>
<th>( \ldots )</th>
<th>( g_T )</th>
</tr>
</thead>
<tbody>
<tr>
<td>((x_1, y_1))</td>
<td>( \tilde{\mathcal{D}}_1 )</td>
<td>( \ast )</td>
<td>( \tilde{\mathcal{D}}_3 )</td>
<td>( \tilde{\mathcal{D}}_T )</td>
<td></td>
</tr>
<tr>
<td>((x_2, y_2))</td>
<td>( \ast )</td>
<td>( \ast )</td>
<td>( \tilde{\mathcal{D}}_3 )</td>
<td>( \tilde{\mathcal{D}}_T )</td>
<td></td>
</tr>
<tr>
<td>((x_3, y_3))</td>
<td>( \ast )</td>
<td>( \tilde{\mathcal{D}}_1 )</td>
<td>( \ast )</td>
<td>( \tilde{\mathcal{D}}_T )</td>
<td></td>
</tr>
<tr>
<td>( \ldots )</td>
<td>( \ast )</td>
<td>( \ast )</td>
<td>( \ast )</td>
<td>( \ast )</td>
<td></td>
</tr>
<tr>
<td>((x_N, y_N))</td>
<td>( \tilde{\mathcal{D}}_1 )</td>
<td>( \tilde{\mathcal{D}}_2 )</td>
<td>( \ast )</td>
<td>( \ast )</td>
<td></td>
</tr>
</tbody>
</table>

\( \ast \): not used for obtaining \( g_t \)
—called out-of-bag (OOB) examples
Number of OOB Examples

**OOB (in ⋄) ⇐⇒ not sampled after** $N$ **drawings**

- probability for $(x_n, y_n)$ to be OOB for $g_t$: $(1 - \frac{1}{N})^N$
- if $N$ large:

$$
\left(1 - \frac{1}{N}\right)^N = \frac{1}{\left(\frac{N}{N-1}\right)^N} = \frac{1}{\left(1 + \frac{1}{N-1}\right)^N} \approx \frac{1}{e}
$$

**OOB size per** $g_t \approx \frac{1}{e} N$
Random Forest

Out-Of-Bag Estimate

OOB versus Validation

**OOB**

<table>
<thead>
<tr>
<th></th>
<th>$g_1$</th>
<th>$g_2$</th>
<th>$g_3$</th>
<th>...</th>
<th>$g_T$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(x_1, y_1)$</td>
<td>$\tilde{D}_1$</td>
<td>$\star$</td>
<td>$\tilde{D}_3$</td>
<td>$\tilde{D}_T$</td>
<td></td>
</tr>
<tr>
<td>$(x_2, y_2)$</td>
<td>$\star$</td>
<td>$\star$</td>
<td>$\tilde{D}_3$</td>
<td>$\tilde{D}_T$</td>
<td></td>
</tr>
<tr>
<td>$(x_3, y_3)$</td>
<td>$\star$</td>
<td>$\tilde{D}_1$</td>
<td>$\star$</td>
<td>$\tilde{D}_T$</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$(x_N, y_N)$</td>
<td>$\tilde{D}_1$</td>
<td>$\tilde{D}_2$</td>
<td>$\star$</td>
<td>$\star$</td>
<td></td>
</tr>
</tbody>
</table>

**Validation**

<table>
<thead>
<tr>
<th></th>
<th>$g_1^-$</th>
<th>$g_2^-$</th>
<th>...</th>
<th>$g_M^-$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{D}_{\text{train}}$</td>
<td>$\mathcal{D}_{\text{train}}$</td>
<td>$\mathcal{D}_{\text{train}}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mathcal{D}_{\text{val}}$</td>
<td>$\mathcal{D}_{\text{val}}$</td>
<td>$\mathcal{D}_{\text{val}}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mathcal{D}_{\text{val}}$</td>
<td>$\mathcal{D}_{\text{val}}$</td>
<td>$\mathcal{D}_{\text{val}}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mathcal{D}_{\text{train}}$</td>
<td>$\mathcal{D}_{\text{train}}$</td>
<td>$\mathcal{D}_{\text{train}}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- $\star$ like $\mathcal{D}_{\text{val}}$: ‘enough’ random examples unused during training
- use $\star$ to validate $g_t$? easy, but rarely needed (why?)
- use $\star$ to validate $G$? $E_{\text{oob}}(G) = \frac{1}{N} \sum_{n=1}^{N} \text{err}(y_n, G^-_n(x_n))$, with $G^-_n$ contains only trees that $x_n$ is OOB of

$E_{\text{oob}}$: self-validation of bagging/RF
Model Selection by OOB Error

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

$$g_m^* = A_m^*(\mathcal{D})$$
$$m^* = \arg\min_{1 \leq m \leq M} E_m$$
$$E_m = E_{\text{val}}(A_m(\mathcal{D}_{\text{train}}))$$

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

$$g_m^* = RF_m^*(\mathcal{D})$$
$$m^* = \arg\min_{1 \leq m \leq M} E_m$$
$$E_m = E_{\text{oob}}(RF_m(\mathcal{D}))$$

- use $E_{\text{oob}}$ for self-validation
- no re-training needed

$E_{\text{oob}}$ often accurate in practice
Feature Selection

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

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

and only ‘learn’ a subset-transform $\Phi(\mathbf{x}) = (x_{i_1}, x_{i_2}, x_{i_{d'}})$ with $d' < d$ for the final hypothesis $g(\Phi(\mathbf{x}))$

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

Hsuan-Tien Lin (NTU CSIE)
Feature Selection by Importance

idea: if possible to estimate

importance(i) for i = 1, 2, . . . , d

then can select i_1, i_2, . . . , i_{d'} of top-d' importance values

Linear Model

\[ s = \mathbf{w}^T \mathbf{x} = \sum_{i=1}^{d} w_i x_i \]

• intuitive estimate: importance(i) = |w_i| with some ‘good’ \( \mathbf{w} \)
• ‘good’ \( \mathbf{w} \): learned with full data
• non-linear models? often not easy

next: feature selection in RF
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 $\{x_{n,i}\}_{n=1}^{N}$): $P(x_i) \approx$ remained

- permutation test:

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

with $\mathcal{D}_p$ containing permuted $\{x_{n,i}\}_{n=1}^{N}$

permutation test: a general statistical tool that can be used for arbitrary non-linear models like RF
Feature Importance in Original Random Forest

**permutation test:**

\[
\text{importance}(i) = \text{performance}(\mathcal{D}) - \text{performance}(\mathcal{D}_p)
\]

with \(\mathcal{D}_p\) containing permuted \(\{x_{n,i}\}_{n=1}^{N}\)

- calculating performance needs re-training and validating on each \(\mathcal{D}_p\) in general
- how to ‘escape’ validation? OOB in RF
- original RF solution:

\[
\text{importance}(i) = E_{oob}(G, \mathcal{D}) - E_{oob}(G, \mathcal{D}_p)
\]

with \(\mathcal{D}_p\) ‘dynamically’ containing permuted \(\{x_{n,i} : n \text{ OOB}\}\) for \(g_t\)

original RF solution often efficient and promising in practice
Fun Time
Lecture 10: Random Forest

- Basic Random Forest Algorithm
- Out-Of-Bag Estimate
- Feature Selection