Decision Trees
Boosting and bagging

CS780/880 Introduction to Machine Learning

04/06/2017
SVM: Classification with Maximum Margin Hyperplane
Kernel SVM: Polynomial and Radial Kernels
Regression Methods

- Covered 6+ classification methods
Regression Methods

- Covered 6+ classification methods
- Regression methods (4+)?
Regression Methods

- Covered 6+ classification methods
- Regression methods (4+)?
- Which ones are generative/discriminative?
Regression Trees

- Predict Baseball Salary based on Years played and Hits
- Example:

```
<table>
<thead>
<tr>
<th>Years &lt; 4.5</th>
<th>Hits &lt; 117.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.11</td>
<td>5.11</td>
</tr>
<tr>
<td>6.00</td>
<td>6.00</td>
</tr>
<tr>
<td>6.74</td>
<td>6.74</td>
</tr>
</tbody>
</table>
```
<table>
<thead>
<tr>
<th>Years</th>
<th>Hits</th>
<th>5.11</th>
<th>6.00</th>
<th>6.74</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Advantages/Disadvantages of Decision Trees

Advantages:
- Interpretability
- Non-linearity
- Little data preparation, scale invariance
- Works with qualitative and quantitative features

Disadvantages:
- Hard to encode prior knowledge
- Difficult to fit
- Limited generalization
Advantages/Disadvantages of Decision Trees

- Advantages:
  - Interpretability
  - Non-linearity
  - Little data preparation, scale invariance
  - Works with qualitative and quantitative features

- Disadvantages:
  - Hard to encode prior knowledge
  - Difficult to fit
  - Limited generalization
Advantages/Disadvantages of Decision Trees

- **Advantages:**
  - Interpretability
  - Non-linearity
  - Little data preparation, scale invariance
  - Works with qualitative and quantitative features

- **Disadvantages:**
  - Hard to encode prior knowledge
  - Difficult to fit
  - Limited generalization
Decision Tree Terminology

- Internal nodes
- Branches
- Leaves

```
Years < 4.5
| Years < 4.5
| Hits < 117.5
| 5.11
| 6.00 6.74
```
Types of Decision Trees

- Regression trees
- Classification tree
Learning a Decision Tree

▶ NP Hard problem
Learning a Decision Tree

- NP Hard problem

- Approximate algorithms (heuristics):
  - ID3, C4.5, C5.0 (classification)
  - CART (classification and regression trees)
  - MARS (regression trees)
  - …
CART: Learning Regression Trees

Two basic steps:

1. Divide predictor space into regions $R_1, \ldots, R_J$

2. Make the same prediction for all data points that fall in $R_j$
CART: Recursive Binary Splitting

- Greedy top-to-bottom approach

- Recursively divide regions to minimize RSS

\[
\sum_{x_i \in R_1} (y_i - \bar{y}_1)^2 + \sum_{x_i \in R_2} (y_i - \bar{y}_2)^2
\]
CART: Splitting Example

\[ X_2 \]

\[ X_1 \]

\[ X_3 \]

\[ X_4 \leq t_1 \]

\[ X_2 \leq t_2 \]

\[ X_1 \leq t_3 \]

\[ X_2 \leq t_4 \]
Tree Pruning

- Bias-variance trade-off with regression trees?
Tree Pruning

- Bias-variance trade-off with regression trees?
- May overfit with many leaves.
Tree Pruning

- Bias-variance trade-off with regression trees?
- May overfit with many leaves.
- Better to build a large tree and then prune it to minimize:

\[
\sum_{m=1}^{\left| T \right|} \sum_{x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha|T|
\]
Tree Pruning

- Bias-variance trade-off with regression trees?
- May overfit with many leaves.
- Better to build a large tree and then prune it to minimize:

\[
\sum_{m=1}^{\mid T \mid} \sum_{x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha \mid T \mid
\]

- Why is it better to prune than to stop early?
Pruning Example
Impact of Pruning

![Graph showing the impact of pruning on mean squared error across different tree sizes for training, cross-validation, and test data. The graph illustrates the reduction in mean squared error as tree size increases, with error bars indicating variability.](image_url)
Classification Trees

▶ Similar to regression trees
▶ Except RSS does not make sense
▶ Use other measures of quality:
  1. Classification error rate
     \[
     1 - \max_k p_{mk}
     \]
     Often too pessimistic in practice
  2. Gini (impurity) index (CART):
     \[
     \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})
     \]
  3. Cross-entropy (information gain) (ID3, C4.5):
     \[
     - \sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}
     \]
▶ ID3, C4.5 do not prune
Why Not Use Classification Error?

Decision tree with classification error

Classification Error

Source: https://sebastianraschka.com/faq/docs/decisiontree-error-vs-entropy.html
Why Not Use Classification Error?

Decision tree with information gain

Entropy

Source: https://sebastianraschka.com/faq/docs/decisiontree-error-vs-entropy.html
Why Not Use Classification Error?

Entropy is more optimistic

Source: https://sebastianraschka.com/faq/docs/decisiontree-error-vs-entropy.html
Pruning in Classification Trees

![Pruning in Classification Trees Diagram]

- **Thal:a**
- **Ca < 0.5**
- **MaxHR < 161.5**
- **RestBP < 157**
- **Chol < 244**
- **MaxHR < 156**
- **MaxHR < 145.5**
- **ChestPain:bc**
- **Chol < 244**
- **Sex < 0.5**
- **Slope < 1.5**
- **Age < 52**
- **Thal:b**
- **Oldpeak < 1.1**
- **RestECG < 1**

<table>
<thead>
<tr>
<th>5</th>
<th>10</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.1</td>
<td>0.2</td>
</tr>
</tbody>
</table>

**Tree Size**  
**Error**  
- **Training**  
- **Cross-Validation**  
- **Test**
Trees vs. Linear Models
Trees vs. KNN

- Trees do not require a distance metric.
- Trees work well with categorical predictors.
- Trees work well in large dimensions.
- KNN are better in low-dimensional problems with complex decision boundaries.
Trees vs. KNN

- Trees do not require a distance metric
- Trees work well with categorical predictors
- Trees work well in large dimensions
- KNN are better in low-dimensional problems with complex decision boundaries
Bagging and Boosting

- Methods for reducing variance of decision trees
- Make predictions using a \textit{weighted vote} of multiple trees
- Boosted trees are some of the most successful general machine learning methods (on Kaggle)
Bagging and Boosting

- Methods for reducing variance of decision trees
- Make predictions using a *weighted vote* of multiple trees
- Boosted trees are some of the most successful general machine learning methods (on Kaggle)

- Disadvantage of using votes of multiple trees?
Bagging

- Stands for “Bootstrap Aggregating”
- Construct multiple bootstrapped training sets:
  \[ T_1, T_2, \ldots, T_B \]
- Fit a tree to each one:
  \[ \hat{f}_1, \hat{f}_2, \ldots, \hat{f}_B \]
- Make predictions by averaging individual tree predictions
  \[
  \hat{f}(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}_b(x)
  \]
- Large values of \( B \) are not likely to overfit, \( B \approx 100 \) is a good choice
Random Forests

- Many trees in bagging will be similar
- Algorithms choose the same features to split on
- Random forests help to address similarity:
  - At each split, choose only from $m$ randomly sampled features
- Good empirical choice is $m = \sqrt{p}$
Cross-validation and Bagging

- No need for cross-validation when bagging
- Evaluating trees on out-of-bag samples is sufficient
Boosting (Gradient Boosting, AdaBoost)

What Kaggle has to say:

source:
Gradient Boosting (Regression)

- Boosting uses all of data, not a random subset (usually)
Gradient Boosting (Regression)

- Boosting uses all of data, not a random subset (usually)
- Also builds trees $\hat{f}_1, \hat{f}_2, \ldots$
Gradient Boosting (Regression)

- Boosting uses all of data, not a random subset (usually)
- Also builds trees $\hat{f}_1, \hat{f}_2, \ldots$
- and weights $\lambda_1, \lambda_2, \ldots$
Gradient Boosting (Regression)

- Boosting uses all of data, not a random subset (usually)
- Also builds trees \( \hat{f}_1, \hat{f}_2, \ldots \)
- and weights \( \lambda_1, \lambda_2, \ldots \)
- Combined prediction:

\[
\hat{f}(x) = \sum_{i} \lambda_i \hat{f}_i(x)
\]
Gradient Boosting (Regression)

- Boosting uses all of data, not a random subset (usually)
- Also builds trees \( \hat{f}_1, \hat{f}_2, \ldots \)
- and weights \( \lambda_1, \lambda_2, \ldots \)
- Combined prediction:

\[
\hat{f}(x) = \sum_{i} \lambda_i \hat{f}_i(x)
\]

- Assume we have \( 1 \ldots m \) trees and weights, next best tree?
Gradient Boosting (Regression)

- Just use gradient descent
Gradient Boosting (Regression)

- Just use gradient descent
- **Objective** is to minimize RSS (1/2):

\[
\frac{1}{2} \sum_{i=1}^{n} (y_i - f(x_i))^2
\]
Gradient Boosting (Regression)

- Just use gradient descent
- **Objective** is to minimize RSS (1/2):

\[
\frac{1}{2} \sum_{i=1}^{n} (y_i - f(x_i))^2
\]

- **Objective** with the new tree \(m + 1\):

\[
\frac{1}{2} \sum_{i=1}^{n} \left( y_i - \sum_{j=1}^{m} \hat{f}_j(x_i) - \hat{f}_{m+1}(x_i) \right)^2
\]
Gradient Boosting (Regression)

- Just use gradient descent
- **Objective** is to minimize RSS (1/2):

\[
\frac{1}{2} \sum_{i=1}^{n} (y_i - f(x_i))^2
\]

- **Objective** with the new tree \( m + 1 \):

\[
\frac{1}{2} \sum_{i=1}^{n} \left( y_i - \sum_{j=1}^{m} \hat{f}_j(x_i) - \hat{f}_{m+1}(x_i) \right)^2
\]

- Greatest reduction in RSS: gradient

\[
y_i - \sum_{j=1}^{m} \hat{f}_j(x_i) \approx \hat{f}_{m+1}(x_i)
\]
Gradient Boosting

- Greatest reduction in RSS: \textit{gradient}

\[
y_i - \sum_{j=1}^{m} \hat{f}_j(x_i) \approx \hat{f}_{m+1}(x_i)
\]
Gradient Boosting

- Greatest reduction in RSS: \textit{gradient}

\[ y_i - \sum_{j=1}^{m} \hat{f}_j(x_i) \approx \hat{f}_{m+1}(x_i) \]

- \textbf{Fit new tree to the following target} (instead of \( y_i \))

\[ y_i - \sum_{j=1}^{m} \hat{f}_j(x_i) \]
Gradient Boosting

- **Greatest reduction in RSS:** gradient

  \[ y_i - \sum_{j=1}^{m} \hat{f}_j(x_i) \approx \hat{f}_{m+1}(x_i) \]

- **Fit new tree to the following target** (instead of \( y_i \))

  \[ y_i - \sum_{j=1}^{m} \hat{f}_j(x_i) \]

- **Compute the weight \( \lambda \) by line search**
Gradient Boosting

- Greatest reduction in RSS: gradient
  \[ y_i - \sum_{j=1}^{m} \hat{f}_j(x_i) \approx \hat{f}_{m+1}(x_i) \]

- Fit new tree to the following target (instead of \( y_i \))
  \[ y_i - \sum_{j=1}^{m} \hat{f}_j(x_i) \]

- Compute the weight \( \lambda \) by line search

- And many other bells and whistles
Scalable and flexible gradient boosting

Interfaces for many languages and environments