Lecture 17: Decision Tree / Random Forest / Ensemble

Where are we? ➔

Five major sections of this course

- Regression (supervised)
- Classification (supervised)
- Unsupervised models
- Learning theory
- Graphical models
Choosing the right estimator

**Scikit-learn**: Regression

Linear model fitted by minimizing a regularized empirical loss with SGD

- SGD Regressor
- Lasso
- ElasticNet
- SVR(kernel='rbf')
- Ensemble Regressors
- Ridge Regression
- SVR(kernel='linear')
Scikit-learn: Classification

To combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability / robustness over a single estimator. (1) averaging / bagging (2) boosting

Next after classification?
Today

- Decision Tree (DT):
  - Tree representation
- Brief information theory
- Learning decision trees
- Bagging
- Random forests: Ensemble of DT
- More about ensemble

A study comparing Classifiers

An Empirical Comparison of Supervised Learning Algorithms

Rich Caruana
Alexandru Niculescu-Mizil
Department of Computer Science, Cornell University, Ithaca, NY 14853 USA

Abstract

A number of supervised learning methods have been introduced in the last decade. Unfortunately, the last comprehensive empirical evaluation of supervised learning was the Statlog Project in the early 90's. We present a large-scale empirical comparison between ten supervised learning methods: SVMs, neural nets, logistic regression, naive bayes, memory-based learning, random forests, decision trees, bagged trees, boosted trees, and boosted stumps. We also examine the effect that calibrating the models via Platt Scaling and Isotonic Regression has on their performance. An important aspect of our study is

A study comparing Classifiers

11 binary classification problems / 8 metrics

Where are we? ➔

Three major sections for classification

- We can divide the large variety of classification approaches into roughly three major types

1. Discriminative
   - directly estimate a decision rule/boundary
   - e.g., logistic regression, support vector machine, decisionTree

2. Generative:
   - build a generative statistical model
   - e.g., naïve bayes classifier, Bayesian networks

3. Instance based classifiers
   - Use observation directly (no models)
   - e.g. K nearest neighbors
A Dataset for classification

\[ f : X \rightarrow C \]

Output as Discrete Class Label
\( C_1, C_2, \ldots, C_L \)

- Data/points/instances/examples/samples/records: \([\text{rows}]\)
- Features/attributes/dimensions/independent variables/covariates/predictors/regressors: \([\text{columns, except the last}]\)
- Target/outcome/response/label/dependent variable: special column to be predicted \([\text{last column}]\)

Example

**Example: Play Tennis**

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D6</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D7</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D8</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D9</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D11</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D12</td>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D13</td>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D14</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>

\(10/28/15\)

Dr. Yanjun Qi / UVA CS 6316 / f15
Anatomy of a decision tree

Each node is a test on one feature/attribute

Outlook
  sunny
  overcast
  rain

Possible attribute values of the node

Humidity
  high
  normal

Yes
No

Windy
  true
  false

Leaves are the decisions

Sample size

Your data gets smaller

Anatomy of a decision tree

Each node is a test on one attribute

Outlook
  sunny
  overcast
  rain

Possible attribute values of the node

Humidity
  high
  normal

Yes
No

Windy
  true
  false

Leaves are the decisions
Apply Model to Test Data:
To ‘play tennis’ or not.

A new test example:
(Outlook==rain) and
(Windy==false)

Pass it on the tree
-> Decision is yes.

(Outlook==overcast) -> yes
(Outlook==rain) and (Windy==false) ->yes
(Outlook==sunny) and (Humidity=normal) ->yes

Three cases of “YES”
Decision trees

- Decision trees represent a disjunction of conjunctions of constraints on the attribute values of instances.

- \(\text{Outlook} == \text{overcast}\)
- OR
- \(((\text{Outlook} == \text{rain}) \text{ and } (\text{Windy} == \text{false}))\)
- OR
- \(((\text{Outlook} == \text{sunny}) \text{ and } (\text{Humidity} == \text{normal}))\)
- \(\Rightarrow\) yes play tennis

Representation

\(Y = ((A \text{ and } B) \text{ or } ((\text{not } A) \text{ and } C))\)
Same concept / different representation

\[ Y = ((A \text{ and } B) \text{ or } ((\neg A) \text{ and } C)) \]

Which attribute to select for splitting?

The distribution of each class (not attribute)

This is bad splitting...
How do we choose which attribute to split?

Which attribute should be used as the test?

Intuitively, you would prefer the one that *separates* the training examples as much as possible.

---

Today

- Decision Tree (DT):
  - Tree representation
  - Brief information theory
  - Learning decision trees
  - Bagging
  - Random forests: Ensemble of DT
  - More about ensemble
Information gain is one criteria to decide on which attribute for splitting

• Imagine:
  – 1. Someone is about to tell you your own name
  – 2. You are about to observe the outcome of a dice roll
  – 2. You are about to observe the outcome of a coin flip
  – 3. You are about to observe the outcome of a biased coin flip

• Each situation have a different amount of uncertainty as to what outcome you will observe.

Information

• Information:
  • ➔ Reduction in uncertainty (amount of surprise in the outcome)

\[ I(E) = \log_2 \frac{1}{p(x)} = -\log_2 p(x) \]

If the probability of this event happening is small and it happens, the information is large.

- Observing the outcome of a coin flip is head \[ I = -\log_2 1/2 = 1 \]
- Observe the outcome of a dice is 6 \[ I = -\log_2 1/6 = 2.58 \]
Entropy

- The *expected amount of information* when observing the output of a random variable $X$

$$H(X) = E(I(X)) = \sum_i p(x_i)I(x_i) = \sum_i p(x_i) \log_2 p(x_i)$$

If the $X$ can have 8 outcomes and all are equally likely

$$H(X) = -\sum_i 1/8 \log_2 1/8 = 3$$

- If there are $k$ possible outcomes

$$H(X) \leq \log_2 k$$

- Equality holds when all outcomes are equally likely

- The more the probability distribution deviates from uniformity, the lower the entropy

e.g. for a random binary variable
Entropy Lower $\Rightarrow$ better purity

- Entropy measures the purity

\[
\begin{align*}
4 + 4 - \\
8 + 0 -
\end{align*}
\]

The distribution is less uniform

- Entropy is lower
- The node is purer

Information gain

- IG(X,Y)=H(Y)-H(Y|X)

Reduction in uncertainty of Y by knowing a feature variable X

Information gain:
= (information before split) – (information after split)
= entropy(parent) – [average entropy(children)]

For IG, the higher, the better
Conditional entropy

\[
H(Y) = - \sum_{i} p(y_i) \log_2 p(y_i)
\]

\[
H(Y \mid X = x_j) = - \sum_{i} p(y_i \mid x_j) \log_2 p(y_i \mid x_j)
\]

\[
H(Y \mid X) = \sum_{j} p(x_j) H(Y \mid X = x_j)
\]

\[
= - \sum_{j} p(x_j) \sum_{i} p(y_i \mid x_j) \log_2 p(y_i \mid x_j)
\]

Example

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Labels</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>X2</td>
<td>Y</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>+</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>+</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>-</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>+</td>
</tr>
</tbody>
</table>

\[
\text{Count}
\]

\[
\text{IG}(X1, Y) = H(Y) - H(Y \mid X1)
\]

\[
H(Y) = - (5/10) \log(5/10) - 5/10 \log(5/10) = 1
\]

\[
H(Y \mid X1) = P(X1=\text{T})H(Y \mid X1=\text{T}) + P(X1=\text{F})H(Y \mid X1=\text{F})
\]

\[
= 4/10 (1 \log 1 + 0 \log 0) + 6/10 (5/6 \log 5/6 + 1/6 \log 1/6)
\]

\[
= 0.39
\]

Information gain (X1, Y) = 1 - 0.39 = 0.61

Which one do we choose

X1 or X2?

\[
H(Y) = - \sum_{i} p(y_i) \log_2 p(y_i)
\]

\[
\{ p(y=+) = 5/10 \quad + \quad 5 \}
\]

\[
\{ p(y=-) = 5/10 \quad - \quad 5 \}
\]

10/28/15
Which one do we choose?

<table>
<thead>
<tr>
<th>X1</th>
<th>X2</th>
<th>Y</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>+</td>
<td>2</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>+</td>
<td>2</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>-</td>
<td>5</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>+</td>
<td>1</td>
</tr>
</tbody>
</table>

Information gain (X1,Y) = 0.61
Information gain (X2,Y) = 0.12

Pick the variable which provides the most information gain about Y

⇒ Then recursively choose next Xi on branches
Decision Trees

- **Caveats**: The number of possible values influences the information gain.
  - The more possible values, the higher the gain (the more likely it is to form small, but pure partitions)

- **Other Purity (diversity) measures**
  - Information Gain
  - Gini (population diversity) \( \sum_{k=1}^{K} \hat{p}_{mk} (1 - \hat{p}_{mk}) \)
    - where \( p_{mk} \) is proportion of class \( k \) at node \( m \)
  - Chi-square Test

---

Overfitting

- You can perfectly fit DT to any training data

- **Instability of Trees**
  - High variance
    - High variance (small changes in training set will result in changes of tree model)
    - Hierarchical structure ➔ Error in top split propagates down

- **Two approaches**:
  - 1. Stop growing the tree when further splitting the data does not yield an improvement
  - 2. Grow a full tree, then prune the tree, by eliminating nodes.
Classification and Regression Trees (CART)

- Partition feature space into set of rectangles
- Fit simple model in each partition

**Summary: Decision trees**

- Non-linear classifier
- Easy to use
- Easy to interpret
- Susceptible to overfitting but can be avoided.
Decision Tree / Random Forest

- **Task**
- **Representation**
- **Score Function**
- **Search/Optimization**
- **Models, Parameters**

**Classification**
- Partition feature space into set of rectangles, local smoothness
- Greedy to find partitions
- Split Purity measure / e.g. IG / cross-entropy / Gini /
- Tree Model (s), i.e. space partition

---

**Today**

- Decision Tree (DT):
  - Tree representation
  - Brief information theory
  - Learning decision trees
  - Bagging
  - Random forests: Ensemble of DT
  - More about ensemble
Bagging

- Bagging or *bootstrap aggregation*
  - a technique for reducing the variance of an estimated prediction function.

- For instance, for classification, a *committee of trees*
  - Each tree casts a vote for the predicted class.

Bootstrap

The basic idea:

randomly draw datasets *with replacement (i.e. allows duplicates)* from the training data, each sample *the same size as the original training set*
With vs Without Replacement

• Bootstrap with replacement can keep the sampling size the same as the original size for every repeated sampling. The sampled data groups are independent on each other.

• Bootstrap without replacement cannot keep the sampling size the same as the original size for every repeated sampling. The sampled data groups are dependent on each other.

Bagging

Create bootstrap samples from the training data
Bagging of DT Classifiers

- N examples
- M features
- Take the majority vote
  i.e. Refit the model to each bootstrap dataset, and then examine the behavior over the B replications.

Bagging for Classification with 0,1 Loss

- Classification with 0, 1 loss
  - Bagging a good classifier can make it better.
  - Bagging a bad classifier can make it worse.
  - Can understand the bagging effect in terms of a consensus of independent weak leaners and wisdom of crowds
Peculiarities

- Model Instability is good when bagging
  - The more variable (unstable) the basic model is, the more improvement can potentially be obtained
  - Low-Variability methods (e.g. LDA) improve less than High-Variability methods (e.g. decision trees)

- Load of Redundancy
  - Most predictors do roughly “the same thing”

Bagging: an simulated example

N = 30 training samples, two classes and p = 5 features, Each feature N(0, 1) distribution and pairwise correlation .95 Response Y generated according to:

\[
\Pr(Y = 1|x_1 \leq 0.5) = 0.2 \quad \Pr(Y = 1|x_1 > 0.5) = 0.8
\]

Test sample size of 2000 Fit classification trees to training set and bootstrap samples B = 200

ESL book / Example 8.7.1
Notice the bootstrap trees are different than the original tree.

Five features highly correlated with each other:

- No clear difference with picking up which feature to split.
- Small changes in the training set will result in different tree.
- But these trees are actually quite similar for classification.

ESL book / Example 8.7.1

Consensus: Majority vote
Probability: Average distribution at terminal nodes

ESL book / Example 8.7.1
Bagging

- Slightly increases model space
  - Cannot help where greater enlargement of space is needed

- Bagged trees are correlated
  - Use random forest to reduce correlation between trees

Today

- Decision Tree (DT):
  - Tree representation
  - Brief information theory
  - Learning decision trees
  - Bagging
  - Random forests: special ensemble of DT
  - More about ensemble
Random forest classifier

- Random forest classifier,
  - an extension to bagging
  - which uses de-correlated trees.
Random Forest Classifier

At each node, when choosing the split feature:
- choose only among \( m \leq M \) features.
Random Forests

For each of our $B$ bootstrap samples

Form a tree in the following manner

Given $p$ dimensions, pick $m$ of them

Split only according to these $m$ dimensions

(we will NOT consider the other $p-m$ dimensions)

Repeat the above steps i & ii for each split

Note: we pick a different set of $m$ dimensions for each split on a single tree
Random Forests

Random forest can be viewed as a refinement of bagging with a tweak of **decorrelating** the trees:

At each tree split, a random subset of $m$ features out of all $p$ features is drawn to be considered for splitting.

Some guidelines provided by Breiman, but be careful to choose $m$ based on specific problem:

- $m = p$ amounts to bagging
- $m = p/3$ or $\log_2(p)$ for regression
- $m = \sqrt{p}$ for classification
Why correlated trees are not ideal?

Random Forests try to reduce correlation between the trees.

Why?

Why correlated trees are not ideal?

Assuming each tree has variance $\sigma^2$

If trees are independently identically distributed, then average variance is $\sigma^2/B$
Why correlated trees are not ideal?

Assuming each tree has variance $\sigma^2$

If simply identically distributed, then average variance is

$$\rho \sigma^2 + \frac{1 - \rho}{B} \sigma^2$$

As $B \to \infty$, second term $\to 0$

Thus, the pairwise correlation always affects the variance

---

Why correlated trees are not ideal?

How to deal?

If we reduce $m$ (the number of dimensions we actually consider),

then we reduce the pairwise tree correlation

Thus, variance will be reduced.
Today

- Decision Tree (DT):
  - Tree representation
  - Brief information theory
  - Learning decision trees
  - Bagging
  - Random forests: Ensemble of DT
  - More [ensemble]

10/28/15

63

e.g. Ensembles in practice

[Netflix Prize]

Oct 2006 - 2009

- Training data is a set of users and ratings (1, 2, 3, 4, 5 stars) those users have given to movies.
- Predict what rating a user would give to any movie

- $1 million prize for a 10% improvement over Netflix’s current method (MSE = 0.9514)
Ensemble in practice

Team “Bellkor’s Pragmatic Chaos” defeated the team “ensemble” by submitting just 20 minutes earlier! ➔ 1 million dollar!

<table>
<thead>
<tr>
<th>Rank</th>
<th>Team Name</th>
<th>Best Test Score</th>
<th>% Improvement</th>
<th>Best Submit Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BellKor’s Pragmatic Chaos</td>
<td>0.6567</td>
<td>10.06</td>
<td>2009-07-25 18:18:29</td>
</tr>
<tr>
<td>2</td>
<td>The Ensemble</td>
<td>0.6567</td>
<td>10.06</td>
<td>2009-07-25 18:38:22</td>
</tr>
<tr>
<td>3</td>
<td>Grand Prize Team</td>
<td>0.6562</td>
<td>9.00</td>
<td>2009-07-10 21:24:10</td>
</tr>
<tr>
<td>4</td>
<td>Opera Solutions and Vandelay United</td>
<td>0.6568</td>
<td>9.84</td>
<td>2009-07-10 01:12:31</td>
</tr>
<tr>
<td>5</td>
<td>Vandelay Industries</td>
<td>0.6591</td>
<td>9.81</td>
<td>2009-07-10 00:32:20</td>
</tr>
<tr>
<td>6</td>
<td>PragmaticTheory</td>
<td>0.6594</td>
<td>9.77</td>
<td>2009-06-24 12:05:56</td>
</tr>
<tr>
<td>7</td>
<td>BellKor in BigChaos</td>
<td>0.6601</td>
<td>9.70</td>
<td>2009-06-13 08:14:39</td>
</tr>
<tr>
<td>8</td>
<td>Dace</td>
<td>0.6612</td>
<td>9.59</td>
<td>2009-07-24 17:18:43</td>
</tr>
<tr>
<td>9</td>
<td>Feeds2</td>
<td>0.6622</td>
<td>9.48</td>
<td>2009-07-12 13:11:51</td>
</tr>
<tr>
<td>10</td>
<td>BigChaos</td>
<td>0.6623</td>
<td>9.47</td>
<td>2009-04-07 12:23:59</td>
</tr>
<tr>
<td>11</td>
<td>Opera Solutions</td>
<td>0.6623</td>
<td>9.47</td>
<td>2009-07-24 00:34:07</td>
</tr>
<tr>
<td>12</td>
<td>BellKor</td>
<td>0.6624</td>
<td>9.46</td>
<td>2009-07-25 17:19:11</td>
</tr>
</tbody>
</table>

The ensemble team ➔ blenders of multiple different methods

References

- Prof. Tan, Steinbach, Kumar’s “Introduction to Data Mining” slide
- Dr. Oznur Tastan’s slides about RF and DT