Introduction to Artificial Intelligence
Decision Trees, Random Forest

Janyl Jumadinova
October 19, 2016
Learning

Learning/training: build a classification or regression rule from a set of samples

Prediction: assign a class or value to new samples
Learning

Supervised
  • Learning set = \{(sample \text{ [acquisition]}, \text{class \text{ [expert]}})\}

Unsupervised
  • Learning set = unlabeled samples

Semi-supervised
  • Learning set = some labeled samples + many unlabeled samples
Ensemble learning

Combining weak classifiers (of the same type)...
... in order to produce a strong classifier
  • Condition: diversity among the weak classifiers

Example: Boosting
  • Train each new weak classifier focusing on samples misclassified by previous ones
  • Popular implementation: AdaBoost
    • Weak classifiers: only need to be better than random guess
Classification Formalized

- Observations are classified into two or more classes, represented by a response variable $Y$ taking values $1, 2, \ldots, K$.
- We have a feature vector $X = (X_1, X_2, \ldots, X_p)$, and we hope to build a classification rule $C(X)$ to assign a class label to an individual with feature $X$.
- We have a sample of pairs $(y_i, x_i), i = 1, \ldots, N$. Note that each of the $x_i$ are vectors.
Decision Tree

- **Root node**
  - Entry point to a collection of data
- **Inner nodes (among which the root node)**
  - A question is asked about data
  - One child node per possible answer
- **Leaf nodes**
  - Correspond to the decision to take (or conclusion to make) if reached
Decision Tree

- Represented by a series of binary splits.
- Each internal node represents a value query on one of the variables e.g. “Is $X_3 > 0.4$”. If the answer is “Yes”, go right, else go left.
Decision Tree

- Represented by a series of binary splits.
- Each internal node represents a value query on one of the variables e.g. “Is $X_3 > 0.4$”. If the answer is “Yes”, go right, else go left.
- The terminal nodes are the decision nodes.
- New observations are classified by passing their $X$ down to a terminal node of the tree, and then using majority vote.
Decision Tree

- Can handle huge datasets
- Can handle mixed predictors—quantitative and qualitative
- Easily ignore redundant variables
- Handle missing data elegantly
- Small trees are easy to interpret
- Large trees are hard to interpret
- Often prediction performance is poor
Model Averaging

Classification trees can be simple, but often produce noisy and weak classifiers.

- **Bagging**: Fit many large trees to bootstrap-resampled versions of the training data, and classify by majority vote.

- **Boosting**: Fit many large or small trees to reweighted versions of the training data. Classify by weighted majority vote.

- **Random Forests**: Fancier version of bagging.
Model Averaging

Classification trees can be simple, but often produce noisy and weak classifiers.

- **Bagging**: Fit many large trees to bootstrap-resampled versions of the training data, and classify by majority vote.
- **Boosting**: Fit many large or small trees to reweighted versions of the training data. Classify by weighted majority vote.
- **Random Forests**: Fancier version of bagging.

**In general**  Boosting $\triangleright$ Random Forests $\triangleright$ Bagging $\triangleright$ Single Tree.
Random Forest

- At each tree split, a random sample of $m$ features is drawn, and only those $m$ features are considered for splitting. Typically $m = \sqrt{p}$ or $\log_2 p$, where $p$ is the number of features.
- For each tree grown on a bootstrap sample, the error rate for observations left out of the bootstrap sample is monitored.
Random Forest

- The more often a gene is chosen as a splitter variable, the higher its “Variable Importance” – This can be used to prioritize which genes to select for an assay with limited gene measurements

<table>
<thead>
<tr>
<th>Gene</th>
<th>Var. Imp.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gene A</td>
<td>0.67</td>
</tr>
<tr>
<td>Gene B</td>
<td>0.20</td>
</tr>
<tr>
<td>Gene D</td>
<td>0.13</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Evaluation

- The **precision** is the ratio \( \frac{tp}{tp + fp} \) where \( tp \) is the number of true positives and \( fp \) the number of false positives.

- The precision is intuitively the ability of the classifier not to label as positive a sample that is negative.
Evaluation

- The precision is the ratio $\frac{tp}{(tp+fp)}$ where $tp$ is the number of true positives and $fp$ the number of false positives.
  - The precision is intuitively the ability of the classifier not to label as positive a sample that is negative.

- The recall is the ratio $\frac{tp}{(tp+fn)}$.
  - The recall is intuitively the ability of the classifier to find all the positive samples.
Evaluation

- The **F-beta score** can be interpreted as a weighted harmonic mean of the precision and recall, where an F-beta score reaches its best value at 1 and worst score at 0.
- The F-beta score weights recall more than precision by a factor of beta. $\beta = 1.0$ means recall and precision are equally important.
The F-beta score can be interpreted as a weighted harmonic mean of the precision and recall, where an F-beta score reaches its best value at 1 and worst score at 0.
- The F-beta score weights recall more than precision by a factor of beta. $\beta = 1.0$ means recall and precision are equally important.

- The support is the number of occurrences of each class in the correct target values.
Classification Summary

- **Support Vector Machines (SVMs):**
  - works for linearly separable and linearly inseparable data;
  - works well in a highly dimensional space (text classification);
  - inefficient to train; probably not applicable to most industry scale applications.

- **Random Forest:**
  - handle high dimensional spaces well, as well as the large number of training data; has been shown to outperform others.
No Free Lunch Theorem:

Wolpert (1996) showed that in a noise-free scenario where the loss function is the misclassification rate, if one is interested in off-training-set error, then there are no a priori distinctions between learning algorithms. On average, they are all equivalent.
No Free Lunch Theorem:
Wolpert (1996) showed that in a noise-free scenario where the loss function is the misclassification rate, if one is interested in off-training-set error, then there are no a priori distinctions between learning algorithms. On average, they are all equivalent.

Occam’s Razor principle:
Use the least complicated algorithm that can address your needs and only go for something more complicated if strictly necessary.

“Do we Need Hundreds of Classifiers to Solve Real World Classification Problems?”