What’s better than a tree?
Random Forest

- Can often improve performance of decision tree classifiers using a set of decision trees (a forest)
- Each tree trained on a random subset of training data
- Classify a data instance using all trees
- Combine answers to make classification
  - E.g., vote for most common class
1. Many trees are created using random subsets of features and bootstrapped data.
2. Each tree votes by predicting target class.
3. Votes are tallied to reach the final prediction.
cf. Wisdom of the Crowd

• Statistician Francis Galton observed a 1906 contest to guess the weight of an ox at a country fair that 800 people entered. He discovered that their average guess (1,197lb) was very close to the actual weight (1,198lb).

• When getting human annotations training data for machine learning, standard practice is get ≥ 3 annotations and take majority vote.
Random Forests Benefits

• Decision trees not the strongest modeling approach
• Random forests make them much stronger
• => more robust than a single decision tree
  – Limit overfitting to given dataset
  – Reduce errors due to training data bias
  – Stable performance if some noise added to training data
Bagging

• Idea can be used on any classifier!
• Improve classification by combining classifications of randomly selected training subsets
• Bagging = Bootstrap aggregating
  An ensemble meta-algorithm that can improve stability & accuracy of algorithms for statistical classification and regression
• Helps avoid overfitting
Bagging
Bootstrap Aggregation

Data → Bootstrapped Data → Models → Voting → Outcome

DATA → DATA → DATA

Train model → MODELS

Voting → VOTE → OUTPUT

Chris Albon

Sampled with replacement
Choosing subsets of training data

- Classic bagging: select random subset of training instances with replacement
- Pasting: select random subset of training instances
- Random Subspaces: use all training instances, but with a random subset of features
- Random Patches: random subset of instances and random subset of features
- What’s best? YMMV: depends on problem, training data, algorithm
Examples

• Two examples using Weka
  – UCI Auto mpg prediction dataset
  – UCI Adult income prediction dataset
• RandomForest improves over J48 for the smaller dataset, but not for the larger
• Takeaway: more data is always best
UCI Auto MGP Dataset

Auto MPG Data Set

Abstract: Revised from CMU StatLib library, data concerns city-cycle fuel consumption.

<table>
<thead>
<tr>
<th>Data Set Characteristics:</th>
<th>Multivariate</th>
<th>Number of Instances:</th>
<th>398</th>
<th>Area:</th>
<th>N/A</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute Characteristics:</td>
<td>Categorical, Real</td>
<td>Number of Attributes:</td>
<td>8</td>
<td>Date Donated</td>
<td>1993-07-07</td>
</tr>
<tr>
<td>Associated Tasks:</td>
<td>Regression</td>
<td>Missing Values?</td>
<td>Yes</td>
<td>Number of Web Hits:</td>
<td>430910</td>
</tr>
</tbody>
</table>

Source:

This dataset was taken from the StatLib library which is maintained at Carnegie Mellon University. The dataset was used in the 1983 American Statistical Association Exposition.

Data Set Information:

This dataset is a slightly modified version of the dataset provided in the StatLib library. In line with the use by Ross Quinlan (1993) in predicting the attribute "mpg", 8 of the original instances were removed because they had unknown values for the "mpg" attribute. The original dataset is available in the file "auto-mpg.data-original".

"The data concerns city-cycle fuel consumption in miles per gallon, to be predicted in terms of 3 multivalued discrete and 5 continuous attributes." (Quinlan, 1993)
UCI Auto MGP Dataset (2)

• Data from 1983
• 398 instances
• Predict auto mpg from seven attributes:
  – Number of cylinders
  – Displacement
  – Horsepower
  – Weight
  – Acceleration
  – Model year
  – Country of origin
Choose J48 -C 0.25 -M 2

Use training set

Time taken to build model: 0.01 seconds

--- Evaluation on training set ---

Time taken to test model on training data: 0 seconds

--- Summary ---

Correctly Classified Instances 230 95.8333 %
Incorrectly Classified Instances 10 4.1667 %
Kappa statistic 0.9174
Mean absolute error 0.0453
Root mean squared error 0.1505
Relative absolute error 13.4303 %
Root relative squared error 36.7193 %
Total Number of Instances 240

--- Detailed Accuracy By Class ---

<table>
<thead>
<tr>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
<th>ROC Area</th>
<th>PRC Area</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.987</td>
<td>0.025</td>
<td>0.987</td>
<td>0.987</td>
<td>0.987</td>
<td>0.963</td>
<td>0.998</td>
<td>0.998</td>
<td>1</td>
</tr>
<tr>
<td>0.881</td>
<td>0.015</td>
<td>0.925</td>
<td>0.811</td>
<td>0.902</td>
<td>0.883</td>
<td>0.991</td>
<td>0.953</td>
<td>2</td>
</tr>
<tr>
<td>0.923</td>
<td>0.025</td>
<td>0.878</td>
<td>0.938</td>
<td>0.900</td>
<td>0.880</td>
<td>0.989</td>
<td>0.921</td>
<td>3</td>
</tr>
</tbody>
</table>
| Weighted Avg. | 0.958 | 0.023 | 0.959 | 0.958 | 0.958

--- Confusion Matrix ---

a  b  c  <= classified as
157 1 1  | a = 1
1 37 4  | b = 2
1 2 36  | c = 3
RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1

Time taken to build model: 0.1 seconds

Correctly Classified Instances 240 100 %
Incorrectly Classified Instances 0 0 %
Kappa statistic 1
Mean absolute error 0.0674
Root mean squared error 0.114
Relative absolute error 19.9659 %
Root relative squared error 27.8064 %
Total Number of Instances 240

TP Rate FP Rate Precision Recall F-Measure MCC ROC Area PRC Area Class
1.000 0.000 1.000 1.000 1.000 1.000 1.000 1.000 1
1.000 0.000 1.000 1.000 1.000 1.000 1.000 1.000 2
1.000 0.000 1.000 1.000 1.000 1.000 1.000 1.000 3

Weighted Avg.
1.000 0.000 1.000 1.000 1.000 1.000 1.000 1.000

a b c <- classified as
159 0 0 | a = 1
0 42 0 | b = 2
0 0 39 | c = 3
100% ... Wait, What?

- Results are too good to be true!
- ML results tend to be asymptotic
  - asymptotic lines approach a curve but never touch
- Closer you get to F1=1.0, the harder it is to improve
- What did we do wrong?
Results are too good

• Relatively small dataset allows construction of a DT model that does very well
• Using Random Forest still improves on it
• We trained and tested on the same data!
• Very poor methodology since it overfits to this particular training set
• This training dataset has a separate test data set
  – We can also try 10-fold cross validation
Classifier: J48 -C 0.25 -M 2

Test options:
- Supplied test set

Classifier output:

Size of the tree: 49

Time taken to build model: 0.02 seconds

Time taken to test model on supplied test set: 0 seconds

=== Evaluation on test set ===

Correctly Classified Instances 112 84.8485 %
Incorrectly Classified Instances 20 15.1515 %
Kappa statistic 0.7255
Mean absolute error 0.1198
Root mean squared error 0.2915
Relative absolute error 32.9443 %
Root relative squared error 66.1432 %
Total Number of Instances 132

=== Summary ===

Correctly Classified Instances 112 84.8485 %
Incorrectly Classified Instances 20 15.1515 %
Kappa statistic 0.7255
Mean absolute error 0.1198
Root mean squared error 0.2915
Relative absolute error 32.9443 %
Root relative squared error 66.1432 %
Total Number of Instances 132

=== Detailed Accuracy By Class ===

<table>
<thead>
<tr>
<th>Class</th>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
<th>ROC Area</th>
<th>PRC Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.987</td>
<td>0.127</td>
<td>0.916</td>
<td>0.987</td>
<td>0.950</td>
<td>0.777</td>
<td>0.967</td>
<td>0.962</td>
</tr>
<tr>
<td>2</td>
<td>0.650</td>
<td>0.063</td>
<td>0.650</td>
<td>0.650</td>
<td>0.650</td>
<td>0.588</td>
<td>0.851</td>
<td>0.660</td>
</tr>
<tr>
<td>3</td>
<td>0.657</td>
<td>0.062</td>
<td>0.793</td>
<td>0.657</td>
<td>0.719</td>
<td>0.535</td>
<td>0.887</td>
<td>0.690</td>
</tr>
</tbody>
</table>

Weighted Avg. 0.848 0.100 0.843 0.948 0.843 0.769 0.928 0.844

=== Confusion Matrix ===

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>classed as</th>
</tr>
</thead>
<tbody>
<tr>
<td>76</td>
<td>0</td>
<td>1</td>
<td>a = 1</td>
</tr>
<tr>
<td>2</td>
<td>13</td>
<td>5</td>
<td>b = 2</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>23</td>
<td>c = 3</td>
</tr>
</tbody>
</table>

Status: OK
Classifier

Choose RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1

Test options

- Use training set
- Supplied test set
- Cross-validation, Folds 10
- Percentage split, % 66
- More options...

Classifier output

Running with 100 iterations and base learner:
weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities

Time taken to build model: 0.09 seconds

Time taken to test model on supplied test set: 0.01 seconds

=== Evaluation on test set ===

Correctly Classified Instances 115 87.1212 %
Incorrectly Classified Instances 17 12.8788 %
Kappa statistic 0.7653
Mean absolute error 0.1642
Root mean squared error 0.2605
Relative absolute error 45.1528 %
Root relative squared error 59.0951 %
Total Number of Instances 132

=== Summary ===

=== Detailed Accuracy By Class ===

<table>
<thead>
<tr>
<th></th>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
<th>ROC Area</th>
<th>PRC Area</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>0.974</td>
<td>0.164</td>
<td>0.893</td>
<td>0.974</td>
<td>0.932</td>
<td>0.831</td>
<td>0.986</td>
<td>0.992</td>
<td>1</td>
</tr>
<tr>
<td>b</td>
<td>0.750</td>
<td>0.036</td>
<td>0.789</td>
<td>0.750</td>
<td>0.769</td>
<td>0.730</td>
<td>0.961</td>
<td>0.838</td>
<td>2</td>
</tr>
<tr>
<td>c</td>
<td>0.714</td>
<td>0.041</td>
<td>0.862</td>
<td>0.714</td>
<td>0.781</td>
<td>0.718</td>
<td>0.965</td>
<td>0.910</td>
<td>3</td>
</tr>
</tbody>
</table>

Weighted Avg. 0.871 0.112 0.869 0.871 0.867

=== Confusion Matrix ===

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>---</th>
<th>classified as</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>75</td>
<td>1</td>
<td>1</td>
<td></td>
<td>a = 1</td>
</tr>
<tr>
<td>b</td>
<td>2</td>
<td>15</td>
<td>3</td>
<td></td>
<td>b = 2</td>
</tr>
<tr>
<td>c</td>
<td>7</td>
<td>3</td>
<td>25</td>
<td></td>
<td>c = 3</td>
</tr>
</tbody>
</table>
AUTO MPG Results (2)

• Using an independent test set shows more realistic balanced F1 score of \textbf{.843}
• Using Random Forest raises this to \textbf{.867}
• While the increase is not large, it is probably statistically significant
• F1 scores this high are difficult to increase dramatically
  – Human scores for many tasks are often in this range (i.e. 0.8 – 0.9)
# UCI Adult Dataset

**Abstract:** Predict whether income exceeds $50K/yr based on census data. Also known as "Census Income" dataset.

<table>
<thead>
<tr>
<th>Data Set Characteristics:</th>
<th>Multivariate</th>
<th>Number of Instances:</th>
<th>48842</th>
<th>Area:</th>
<th>Social</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute Characteristics:</td>
<td>Categorical, Integer</td>
<td>Number of Attributes:</td>
<td>14</td>
<td>Date Donated</td>
<td>1996-05-01</td>
</tr>
<tr>
<td>Associated Tasks:</td>
<td>Classification</td>
<td>Missing Values?</td>
<td>Yes</td>
<td>Number of Web Hits:</td>
<td>1470139</td>
</tr>
</tbody>
</table>

**Source:**
Ronny Kohavi and Barry Becker
Data Mining and Visualization
Silicon Graphics.
e-mail: ronnyk@live.com for questions.

**Data Set Information:**
Extraction was done by Barry Becker from the 1994 Census database. A set of reasonably clean records was extracted using the following conditions: 

\[(\text{AGE}>16) \&\& (\text{AGI}>100) \&\& (\text{AFNLWG}T>1) \&\& (\text{HSWK}>0)\]

Prediction task is to determine whether a person makes over 50K a year.

**Attribute Information:**
Listing of attributes:
UCI Adult Dataset (2)

• Data on adults from 1994 census data
• Large dataset with 48,842 instances
• Predict if person makes over $50K/year
  – Equivalent to ~$87K/year today
• 14 features including age, education, marital status, occupation, race, sex, native country, ...
  – Mixture of numeric (e.g., age) and nominal (e.g., occupation) values
Classifier:

```
Choose RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1
```

Test options:

- Use training set

Classifier output:

```
Bagging with 100 iterations and base learner:

weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities

Time taken to build model: 15.17 seconds

Time taken to test model on training set ===

Time taken to test model on training data: 6.52 seconds

Summary ===

Correctly Classified Instances 48774 99.8608 %
Incorrectly Classified Instances 68 0.1392 %
Kappa statistic 0.9962
Mean absolute error 0.0737
Root mean squared error 0.1263
Relative absolute error 20.2565 %
Root relative squared error 29.6022 %
Root number of instances 48842

Detailed Accuracy By Class ===

<table>
<thead>
<tr>
<th></th>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
<th>ROC Area</th>
<th>PRC Area</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;50K</td>
<td>0.995</td>
<td>0.000</td>
<td>1.000</td>
<td>0.995</td>
<td>0.997</td>
<td>0.996</td>
<td>1.000</td>
<td>1.000</td>
<td>&gt;50K</td>
</tr>
<tr>
<td>&lt;=50K</td>
<td>1.000</td>
<td>0.005</td>
<td>0.998</td>
<td>1.000</td>
<td>0.999</td>
<td>0.996</td>
<td>1.000</td>
<td>1.000</td>
<td>&lt;=50K</td>
</tr>
<tr>
<td>Weighted Avg.</td>
<td>0.999</td>
<td>0.004</td>
<td>0.999</td>
<td>0.999</td>
<td>0.999</td>
<td>0.999</td>
<td>1.000</td>
<td>1.000</td>
<td></td>
</tr>
</tbody>
</table>

Confusion Matrix ===

```
a  b  --- classified as
11624 63   a = >50K
5 37150  b = <=50K
```

Status:

OK
Result

- Significant increase on F1 scores when both trained and evaluated on training set
- This is considered to be poor methodology since it overfits to the particular training set
Create train and test collection

• Train has ~95% of data, test 5%
• Trained models for J48 and random forest using train dataset
• Tested on test data set
• Results were that random forest was (at best) about the same as J48
• Large dataset reduced problem of overfitting, so random forest did not help
Classifier

Choose J48 -C 0.25 -M 2

Test options

- Use training set
- Supplied test set: Set...
- Cross-validation: Folds 10
- Percentage split: % 66

More options...

Classifier output

--- Summary ---

Correctly Classified Instances 2155 86.2 %
Incorrectly Classified Instances 345 13.8 %
Kappa statistic 0.5988
Mean absolute error 0.1391
Root mean squared error 0.3196
Relative absolute error 52.5531 %
Root relative squared error 74.1954 %
Total Number of Instances 2500

--- Detailed Accuracy By Class ---

<table>
<thead>
<tr>
<th></th>
<th>TP Rate</th>
<th>FP Rate</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
<th>MCC</th>
<th>ROC Area</th>
<th>PRC Area</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;50K</td>
<td>0.611</td>
<td>0.056</td>
<td>0.780</td>
<td>0.611</td>
<td>0.686</td>
<td>0.606</td>
<td>0.895</td>
<td>0.759</td>
<td>&gt;50K</td>
</tr>
<tr>
<td>&lt;=50K</td>
<td>0.944</td>
<td>0.389</td>
<td>0.880</td>
<td>0.944</td>
<td>0.912</td>
<td>0.606</td>
<td>0.895</td>
<td>0.953</td>
<td>&lt;=50K</td>
</tr>
</tbody>
</table>

Weighted Avg.

0.862 0.307 0.857 0.862 0.856 0.606

--- Confusion Matrix ---

a b  classified as
a 376 239 | a = >50K
b 106 1779 | b = <=50K

Status

OK
Conclusions

• Bagging can help, especially if amount of training data adequate, but not as large as it should be

• While we explore it using decision trees, it can be applied to any classifier
  – Scikit-learn has a general module for bagging

• In general, using any of several ensemble approaches to classification is often very helpful
Conclusions

• Wait, there’s more...

• A classification problem can change over time
  – E.g.: recognizing a spam message from its content and metadata

• We showed that an ensemble approach can detect a change in the nature of spam
  – Which tells us its time to retrain with new data
Recognizing Concept Drift

• Build ensemble of five models to classify spam comments left on a blog at time T1
• Note the relative level of agreement
• Detect when one of the models starts to diverge from the others with at time T2
  – Time to get new data and retrain
  – Examining disagreements can be enlightening
• Used temporal data spanning several years to prove effectiveness
  – E.g., spam moved from viagra to weight loss