Using multiple models:
Bagging, Boosting, Ensembles, Forests
Bagging

• Combining predictions from multiple models
• Different models obtained from bootstrap samples of training data
• Average predictions or majority voting from multiple models

• If different training datasets cause significant differences in learned model, then bagging can improve accuracy.
### Performance (decision tree)

#### Mis-classification Rates (Percent)

<table>
<thead>
<tr>
<th>Data Set</th>
<th>$e_S$</th>
<th>$e_B$</th>
<th>Decrease</th>
</tr>
</thead>
<tbody>
<tr>
<td>Waveform</td>
<td>29.0</td>
<td>19.4</td>
<td>33%</td>
</tr>
<tr>
<td>Heart</td>
<td>10.0</td>
<td>5.3</td>
<td>47%</td>
</tr>
<tr>
<td>breast cancer</td>
<td>6.0</td>
<td>4.2</td>
<td>30%</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>11.2</td>
<td>8.6</td>
<td>23%</td>
</tr>
<tr>
<td>Diabetes</td>
<td>23.4</td>
<td>8.8</td>
<td>20%</td>
</tr>
<tr>
<td>Glass</td>
<td>32.0</td>
<td>24.9</td>
<td>22%</td>
</tr>
<tr>
<td>Soybean</td>
<td>14.5</td>
<td>10.6</td>
<td>27%</td>
</tr>
</tbody>
</table>

50 bootstrap samples each.
## Performance (Regression trees)

Mean Squared Test Set Error

<table>
<thead>
<tr>
<th>Data Set</th>
<th>$e_S$</th>
<th>$e_B$</th>
<th>Decrease</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boston Housing</td>
<td>19.1</td>
<td>11.7</td>
<td>39%</td>
</tr>
<tr>
<td>Ozone</td>
<td>23.1</td>
<td>18.0</td>
<td>22%</td>
</tr>
<tr>
<td>Friedman #1</td>
<td>11.4</td>
<td>6.2</td>
<td>46%</td>
</tr>
<tr>
<td>Friedman #2</td>
<td>30,800</td>
<td>21,700</td>
<td>30%</td>
</tr>
<tr>
<td>Friedman #3</td>
<td>0.0403</td>
<td>0.0249</td>
<td>38%</td>
</tr>
</tbody>
</table>
Bagging

• Advantage – improved accuracy

• Disadvantage – no simple, interpretable model

• Bagging can improve performance for unstable learning algorithms
  – Performance of stable learning methods can deteriorate

• Bagging good models can make them optimal
  – Bagging poor models can make them worse
Boosting

- Multiple models developed in sequence by assigning higher weights (boosting) for those training cases that are difficult to classify

Generate the first model
Repeat
  Weight the training data such that the misclassified cases get higher weights
  Generate the next model
Combine predictions from individual models (weighted by accuracy of the models)
Stream Data Mining

Examples
• Surveillance of trading data for security fraud
• Mining web click stream data
• Monitoring network traffic
• Analysis of sensor network data
• Measuring power consumption

Stream data
• Continuous flow of information vs. finite, statically stored data
• Data volumes too large to be stored on permanent devices
• ‘Burstiness’ in data - data rate of the stream isn't constant
• Continuously evolving patterns
Stream data mining - approaches

- Data condensation/summarization
  - condensed representation is used to track the changes over time
- Temporal granularity
  - Greater importance to more recent data
- Privacy
  - Protecting personal data in the stream data points
  - K-anonymity: minimum k other data points from which this cannot be distinguished

- Clustering
  - Stream data points is first represented as members of micro-clusters; then track these micro-clusters rather than the potentially infinite number of individual data points
- Classifier ensembles
  - Separate classifiers developed on blocks of sequential data, instead of continuously updating a single classifier
  - Each classifier in ensemble can be weighted by accuracy on the current test cases
Random Forests

• Multiple trees developed
  – Majority voting for predicted class - each tree votes for the predicted class label of an example

• Developing each tree
  (N = training data size, M = number of variables)
  – Sample N cases at random, with replacement – use as training data for this tree (bootstrap sample)
  – Select m << M, and use m randomly chosen variables for selection at each node (select the best split amongst the m variables)
  – Grow the full tree – without pruning

(m is a fixed parameter)
Random forests – error rate

• Error rate depends on
  – correlation between any two trees in the forest
    • higher correlation increases the forest error rate
  – error rate of each individual tree in the forest
    • a tree with a low error rate decreases the forest error rate

• Reducing m
  – reduces correlation
  – increases the error rate of individual trees

Optimal m value (usually in a wide range)
Random Forest – error rate

- Out-of-bag error estimate
  - Bootstrapped sample – leaves out about a third of the cases (oob)
  - For each tree, pass each oob case (say, case q) thru’ the tree and get its class prediction (say, j is the predicted class from the tree)
  - Proportion of times that j does not match the true class (out of the number of times that the case q is oob) gives the error rate for this case
  - Average over the error across all training cases

(an unbiased estimate of error)
Random Forests – variable importance

• Importance of a variable v
  – For each tree, run the oob cases thru’ the tree and find the number of correct classifications (nCorrectA)
  – For a variable v, randomly permute its values amongst the oob cases and run the oob cases thru the tree; find the number of correct classifications (nCorrectP)
  – Calculate (nCorrectA – nCorrectP) for the tree
  – Average over all trees gives importance score of variable v

• Significance
  – If importance score is independent for different trees, then z-scores can be calculated and significance level obtained assuming normal distribution of scores
    (low correlations between importance scores across trees for many datasets)
Random Forests - Proximities

• Proximity
  – After each tree is built, pass all the cases thru’ the tree
  – If two cases j and k are in the same terminal node, increase their proximity score by 1
  – Normalize the proximity values - divide by number of trees

• A measure of similarity between two data points
• Can be used for imputing missing values
Random Forests
Missing values using proximity

If variable \( m \) in the \( j \)-th case is missing

• Iterative process to fill missing value
  
  – Weighted average over all cases \( k \) where the variable is not missing, weighted by proximity\((j, k)\)
    (For categorical variable, take the most frequent value, with frequency weighted by proximities)
  
  – Replaced missing values are used in next iteration of RF, where new proximities are calculated
  
  – Re-calculate fills for missing value
  
  – Repeat till no further improvements (4-6 iterations adequate)
Random Forests – missing values replacement

![Graph showing error vs missing for mfixall and mfixtr](image-url)
Random Forests
Detecting outliers using proximities

• Outlier – a case with low proximity to all other cases

• Measure of ‘outlyingness’

For a case j: \(1/(\text{sum of squares of prox}(j,k) \text{ for all other cases } k)\)

Values > 10 are outliers?

• Mis-labeled data can be outliers
  – Class labels can be prone to error
  – Labels often assigned by hand
Outliers based on proximity

PIMA Indian hepatitis data, 768 cases)
Mislabeled cases as outliers
Random Forests
Picturing the Data

• Matrix of proximity values \( \text{Proximity}(j, k) \)

• \( (1 - \text{proximity}(j, k)) \) gives the squared distance in high-dimensional Euclidean space

• Metric scaling – projects the data into lower dimensional space, while preserving the distances between them
  – scaling coordinates (related to the eigenvectors of a modified version of the proximity matrix)
  – 3 or 4 scaling coordinates usually enough
  – Plot 1\(^\text{st}\) versus the 2\(^{\text{nd}}\) scaling coordinates to get a picture of the data
Using proximities and scaling coordinates to picture the data
Random Forests
Unsupervised learning - Clustering

• Data with no class labels
  – How to use to grow trees?

• Original data of N cases – class 1

• Create a synthetic data set of N cases – class 2
  – Random sampling from univariate distributions in original data

  – Let \( x(v, j) \) be the \( v \)-th variable in the \( j \)-th case of class 1
  
  For a case of class 2:
  
  Select the 1\(^{st}\) variable’s value at random from the N values \( x(1, j), j=1,..N \)
  
  Select the 2nd variable’s value at random from the N values \( x(2, j), j=1,..N \)
  
  Etc.

  – Class 2 destroys the dependencies between variables in Class 1
Random Forests - Clustering

• Use RF on 2 class problem
  – Error rates close to 50% implies RF cannot distinguish
  – Low error implies good separation between classes
    • Strong dependency structure amongst variables in the original data
    • We can use all the RF tools (proximities, scaling views, variable importance, outliers, etc.) on the original data

• Testing approach:
  – Dataset with class labels, Remove labels
  – Apply clustering
  – Do clusters correspond to original classes?
Metric scaling for supervised learning data
RF for clustering
Metric scaling for unsupervised learning data

![Graph showing clustering results with three classes:
- Class 1 represented by green triangles
- Class 2 represented by blue plus signs
- Class 3 represented by red triangles
The graph is plotted on a 2D coordinate system with the x-axis labeled '1st Scaling Coordinate' and the y-axis labeled '2nd Scaling Coordinate'.