Lecture 8: Decision Trees and Overfitting

Osbert Bastani and Zachary G. Ives
CIS 4190/5190 – Fall 2022
Tasks

- Homework 2 due October 3 8pm
- Project team member submission: due October 4, 8pm
Recall from Last Time

Two kinds of nonparametric learning: k-Nearest Neighbor and Decision Trees

Decision tree algorithm (C4.5):
- Greedy recursive algorithm: successively splits the training data into “hyperrectangles”

Assume Boolean functions as the basis of intermediate notes
Intermediate note splits are chosen based on information gain

Basic scheme: use entropy as a measure of information gain

\[
H(\mathcal{D}) = - \sum_c P(Y = c) \log_2 P(Y = c), \quad \text{for each class } c
\]

\[
\text{IG}(\mathcal{D}, X_j) = H(\mathcal{D}) - \sum_v H(\mathcal{D}[X_j = v]) P(X_j = v) \quad \text{for each value } v \text{ of } X_j
\]
# Revisiting Diabetes Data

Data from NHANES 2013/14 survey

<table>
<thead>
<tr>
<th>SEQN</th>
<th>RIDAGEYR</th>
<th>BMXWAIST</th>
<th>BMXHT</th>
<th>LBXTC</th>
<th>BMXLEG</th>
<th>BMXWT</th>
<th>BMXBMI</th>
<th>RIDRETH1</th>
<th>BPQ020</th>
<th>ALQ12OQ</th>
<th>DMDEDUC2</th>
<th>RIAGENDR</th>
<th>INDFMPR</th>
<th>LBXGH</th>
<th>DIABETIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>73557</td>
<td>69.0</td>
<td>100.0</td>
<td>171.3</td>
<td>167.0</td>
<td>39.2</td>
<td>78.3</td>
<td>26.7</td>
<td>Non-Hispanic Black</td>
<td>yes</td>
<td>1.0</td>
<td>high school graduate / GED</td>
<td>male</td>
<td>0.84</td>
<td>13.9</td>
<td>yes</td>
</tr>
<tr>
<td>73558</td>
<td>54.0</td>
<td>107.6</td>
<td>176.8</td>
<td>170.0</td>
<td>40.0</td>
<td>89.5</td>
<td>28.6</td>
<td>Non-Hispanic White</td>
<td>yes</td>
<td>7.0</td>
<td>high school graduate / GED</td>
<td>male</td>
<td>1.78</td>
<td>9.1</td>
<td>yes</td>
</tr>
<tr>
<td>73559</td>
<td>72.0</td>
<td>109.2</td>
<td>175.3</td>
<td>126.0</td>
<td>40.0</td>
<td>88.9</td>
<td>28.9</td>
<td>Non-Hispanic White</td>
<td>yes</td>
<td>0.0</td>
<td>some college or AA degree</td>
<td>male</td>
<td>4.51</td>
<td>8.9</td>
<td>yes</td>
</tr>
<tr>
<td>73562</td>
<td>56.0</td>
<td>123.1</td>
<td>158.7</td>
<td>226.0</td>
<td>34.2</td>
<td>105.0</td>
<td>41.7</td>
<td>Mexican American</td>
<td>yes</td>
<td>5.0</td>
<td>some college or AA degree</td>
<td>male</td>
<td>4.79</td>
<td>5.5</td>
<td>no</td>
</tr>
<tr>
<td>73564</td>
<td>61.0</td>
<td>110.8</td>
<td>161.8</td>
<td>168.0</td>
<td>37.1</td>
<td>93.4</td>
<td>35.7</td>
<td>Non-Hispanic White</td>
<td>yes</td>
<td>2.0</td>
<td>college graduate or above</td>
<td>female</td>
<td>5.0</td>
<td>5.5</td>
<td>no</td>
</tr>
<tr>
<td>73566</td>
<td>56.0</td>
<td>85.5</td>
<td>152.8</td>
<td>278.0</td>
<td>32.4</td>
<td>61.8</td>
<td>26.5</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>1.0</td>
<td>high school graduate / GED</td>
<td>female</td>
<td>0.48</td>
<td>5.4</td>
<td>no</td>
</tr>
<tr>
<td>73567</td>
<td>65.0</td>
<td>93.7</td>
<td>172.4</td>
<td>173.0</td>
<td>40.0</td>
<td>65.3</td>
<td>22.0</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>4.0</td>
<td>9th-11th grade</td>
<td>male</td>
<td>1.2</td>
<td>5.2</td>
<td>no</td>
</tr>
<tr>
<td>73568</td>
<td>26.0</td>
<td>73.7</td>
<td>152.5</td>
<td>168.0</td>
<td>34.4</td>
<td>47.1</td>
<td>20.3</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>2.0</td>
<td>college graduate or above</td>
<td>female</td>
<td>5.0</td>
<td>5.2</td>
<td>no</td>
</tr>
<tr>
<td>73571</td>
<td>76.0</td>
<td>122.1</td>
<td>172.5</td>
<td>167.0</td>
<td>35.5</td>
<td>102.4</td>
<td>34.4</td>
<td>Non-Hispanic White</td>
<td>yes</td>
<td>2.0</td>
<td>college graduate or above</td>
<td>male</td>
<td>5.0</td>
<td>6.9</td>
<td>yes</td>
</tr>
<tr>
<td>73577</td>
<td>32.0</td>
<td>100.0</td>
<td>166.2</td>
<td>182.0</td>
<td>36.5</td>
<td>79.7</td>
<td>28.9</td>
<td>Mexican American</td>
<td>no</td>
<td>20.0</td>
<td>Less than 9th grade</td>
<td>male</td>
<td>0.29</td>
<td>5.3</td>
<td>no</td>
</tr>
<tr>
<td>73581</td>
<td>50.0</td>
<td>99.3</td>
<td>185.0</td>
<td>202.0</td>
<td>42.8</td>
<td>80.9</td>
<td>23.6</td>
<td>Other or Multi-Racial</td>
<td>no</td>
<td>0.0</td>
<td>college graduate or above</td>
<td>male</td>
<td>5.0</td>
<td>5.0</td>
<td>no</td>
</tr>
<tr>
<td>73595</td>
<td>28.0</td>
<td>90.3</td>
<td>175.1</td>
<td>198.0</td>
<td>40.5</td>
<td>92.2</td>
<td>30.1</td>
<td>Other or Multi-Racial</td>
<td>no</td>
<td>4.0</td>
<td>some college or AA degree</td>
<td>male</td>
<td>2.26</td>
<td>5.0</td>
<td>no</td>
</tr>
<tr>
<td>73598</td>
<td>35.0</td>
<td>94.6</td>
<td>172.9</td>
<td>192.0</td>
<td>39.1</td>
<td>78.3</td>
<td>26.2</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>2.0</td>
<td>high school graduate / GED</td>
<td>male</td>
<td>1.74</td>
<td>5.5</td>
<td>no</td>
</tr>
<tr>
<td>73599</td>
<td>58.0</td>
<td>114.8</td>
<td>175.3</td>
<td>165.0</td>
<td>40.1</td>
<td>96.0</td>
<td>31.2</td>
<td>Other Hispanic</td>
<td>no</td>
<td>1.0</td>
<td>some college or AA degree</td>
<td>male</td>
<td>3.09</td>
<td>7.7</td>
<td>no</td>
</tr>
<tr>
<td>73566</td>
<td>57.0</td>
<td>117.8</td>
<td>164.7</td>
<td>151.0</td>
<td>35.3</td>
<td>104.0</td>
<td>38.3</td>
<td>Other or Multi-Racial</td>
<td>yes</td>
<td>1.0</td>
<td>college graduate or above</td>
<td>female</td>
<td>5.0</td>
<td>5.9</td>
<td>no</td>
</tr>
<tr>
<td>73600</td>
<td>37.0</td>
<td>122.9</td>
<td>185.1</td>
<td>189.0</td>
<td>48.1</td>
<td>126.2</td>
<td>36.8</td>
<td>Non-Hispanic Black</td>
<td>yes</td>
<td>2.0</td>
<td>high school graduate / GED</td>
<td>male</td>
<td>0.63</td>
<td>6.2</td>
<td>yes</td>
</tr>
<tr>
<td>73604</td>
<td>69.0</td>
<td>96.6</td>
<td>156.9</td>
<td>203.0</td>
<td>37.0</td>
<td>59.5</td>
<td>24.2</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>1.0</td>
<td>some college or AA degree</td>
<td>female</td>
<td>2.44</td>
<td>5.4</td>
<td>no</td>
</tr>
<tr>
<td>73607</td>
<td>75.0</td>
<td>130.5</td>
<td>196.6</td>
<td>161.0</td>
<td>36.5</td>
<td>111.9</td>
<td>38.9</td>
<td>Non-Hispanic White</td>
<td>yes</td>
<td>0.0</td>
<td>high school graduate / GED</td>
<td>male</td>
<td>1.08</td>
<td>5.0</td>
<td>no</td>
</tr>
<tr>
<td>73610</td>
<td>43.0</td>
<td>102.6</td>
<td>176.8</td>
<td>200.0</td>
<td>38.8</td>
<td>90.2</td>
<td>28.9</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>5.0</td>
<td>college graduate or above</td>
<td>male</td>
<td>2.03</td>
<td>4.9</td>
<td>no</td>
</tr>
<tr>
<td>73613</td>
<td>60.0</td>
<td>113.6</td>
<td>163.8</td>
<td>203.0</td>
<td>41.6</td>
<td>104.9</td>
<td>39.1</td>
<td>Non-Hispanic Black</td>
<td>yes</td>
<td>2.0</td>
<td>9th-11th grade</td>
<td>female</td>
<td>5.0</td>
<td>6.1</td>
<td>no</td>
</tr>
<tr>
<td>73614</td>
<td>55.0</td>
<td>90.9</td>
<td>167.9</td>
<td>256.0</td>
<td>43.5</td>
<td>60.9</td>
<td>21.6</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>0.0</td>
<td>high school graduate / GED</td>
<td>female</td>
<td>1.29</td>
<td>5.0</td>
<td>no</td>
</tr>
<tr>
<td>73615</td>
<td>65.0</td>
<td>100.3</td>
<td>145.9</td>
<td>166.0</td>
<td>30.0</td>
<td>55.4</td>
<td>26.0</td>
<td>Other Hispanic</td>
<td>yes</td>
<td>1.0</td>
<td>Less than 9th grade</td>
<td>female</td>
<td>1.22</td>
<td>6.3</td>
<td>yes</td>
</tr>
</tbody>
</table>

© 2019-22 D. Jayaraman, O. Bastani, Z. Ives

Data from NHANES 2013/14 survey
Information Gain Example for Diabetes

First Split

We compared two candidates:

\[ IG(\mathcal{D}, \text{High BP}) = H(\mathcal{D}) - H(\mathcal{D} | \text{High BP}) = 0.918 - 0.459 = 0.459 \]

\[ IG(\mathcal{D}, \text{Education}) = H(\mathcal{D}) - H(\mathcal{D} | \text{Education}) = 0.918 - 0.730 = 0.188 \]
Discounting for Many-Valued Attributes

IG tends toward selecting features that have many values
- e.g., unique identifiers, *dates*, etc.
- unique partitions $\rightarrow$ minimal impurity

Gain Ratio scales by entropy of the sub-dataset proportions:

$$\text{GainRatio}(\mathcal{D}, X_j) = \frac{\text{IG}(\mathcal{D}, X_j)}{\text{SplitInfo}(\mathcal{D}, X_j)}$$

$$\text{SplitInfo}(\mathcal{D}, X_j) = -\sum_v P(X_j = v) \log_2 P(X_j = v)$$

aka Intrinsic Information

This scales by the entropy of the split itself, ignoring the classes
Gain Ratio Example

Already Computed:
- $H(\mathcal{D}) = 0.918$
- $H(\mathcal{D} | \text{High BP}) = 0.459$
- $H(\mathcal{D} | \text{Education}) = 0.730$
- $\text{IG}(\mathcal{D} \text{ High BP}) = 0.459$
- $\text{IG}(\mathcal{D}, \text{Education}) = 0.188$

Need to compute:

$\text{GainRatio}(\mathcal{D}, \text{High BP}) = \frac{\text{IG}(\mathcal{D}, \text{High BP})}{\text{SplitInfo}(\mathcal{D}, \text{High BP})}$

$\text{GainRatio}(\mathcal{D}, \text{Education}) = \frac{\text{IG}(\mathcal{D}, \text{Education})}{\text{SplitInfo}(\mathcal{D}, \text{Education})}$
Gain Ratio Example

Already Computed:
- \( H(\mathcal{D}) = 0.918 \)
- \( H(\mathcal{D} \mid \text{High BP}) = 0.459 \)
- \( H(\mathcal{D} \mid \text{Education}) = 0.730 \)
- \( \text{IG}(\mathcal{D}, \text{High BP}) = 0.459 \)
- \( \text{IG}(\mathcal{D}, \text{Education}) = 0.188 \)

Need to compute:
\[
\text{GainRatio}(\mathcal{D}, \text{High BP}) = \frac{\text{IG}(\mathcal{D}, \text{High BP})}{\text{SplitInfo}(\mathcal{D}, \text{High BP})}
\]
\[
= 0.459
\]
\[
\text{GainRatio}(\mathcal{D}, \text{Education}) = \frac{\text{IG}(\mathcal{D}, \text{Education})}{\text{SplitInfo}(\mathcal{D}, \text{Education})}
\]
\[
= 0.188
\]
Gain Ratio Example

Already Computed:
• $H(\mathcal{D}) = 0.918$
• $H(\mathcal{D} \mid \text{High BP}) = 0.459$
• $H(\mathcal{D} \mid \text{Education}) = 0.730$
• $\text{IG}(\mathcal{D} \mid \text{High BP}) = 0.459$
• $\text{IG}(\mathcal{D}, \text{Education}) = 0.188$

Need to compute:

GainRatio($\mathcal{D}$, High BP) = $\text{IG}(\mathcal{D}, \text{High BP}) / \text{SplitInfo}(\mathcal{D}, \text{High BP})$

GainRatio($\mathcal{D}$, Education) = $\text{IG}(\mathcal{D}, \text{Education}) / \text{SplitInfo}(\mathcal{D}, \text{Education})$

= $-1/12 \log_2 1/12 - 1/12 \log_2 1/12$
  - $3/12 \log_2 3/12 - 3/12 \log_2 3/12$
  - $4/12 \log_2 4/12$

= 2.1258
Gain Ratio Example

Already Computed:
- \( H(\mathcal{D}) = 0.918 \)
- \( H(\mathcal{D} \mid \text{High BP}) = 0.459 \)
- \( H(\mathcal{D} \mid \text{Education}) = 0.730 \)
- \( IG(\mathcal{D} \mid \text{High BP}) = 0.459 \)
- \( IG(\mathcal{D}, \text{Education}) = 0.188 \)

Need to compute:
- \( \text{GainRatio}(\mathcal{D} \mid \text{High BP}) = \frac{IG(\mathcal{D}, \text{High BP})}{\text{SplitInfo}(\mathcal{D}, \text{High BP})} = \frac{0.459}{1} = 0.459 \)
- \( \text{GainRatio}(\mathcal{D}, \text{Education}) = \frac{IG(\mathcal{D}, \text{Education})}{\text{SplitInfo}(\mathcal{D}, \text{Education})} = \frac{0.188}{2.12} = 0.089 \)

Same as before.... But much stronger preference for BP!
Gain Ratio vs “Standard” Information Gain

Gain ratio is **Information Gain** scaled discounted by the **Intrinsic Information** of the split itself.

→ Biases against many-valued splits, which otherwise may have less impurity simply due to size.

Adds a bit of extra computational overhead, so it is not *always* used – but it can be helpful in many real-world use cases!
Another Alternative: The Gini Index

Issue: choosing a split point by IG is a bit expensive – logarithm is an expensive float operation

Popular alternative to entropy: **Gini index**, produces similar results and is less expensive to compute

- Measures how often a randomly chosen element from a set would be incorrectly labeled, if it was **randomly labeled according to the distribution of labels in the subset**
- Like entropy, ranges from 0 – 1 with max value at 50%

\[
Gini(p) = \sum_{i=1}^{K} p_i(1 - p_i) = 1 - \sum_{i=1}^{K} p_i^2
\]

used in one common decision-tree algorithm (CaRT) and in SciKit-Learn

© 2019-22 D. Jayaraman, O. Bastani, Z. Ives
Feature Scaling in Decision Trees

Decision trees are generally univariate -- split one feature (dimension) at a time.

While this limits
They are scale invariant, i.e., we don’t need to standardize the scale!
DT Training for Diabetes
We are Ready to Train the DT for Diabetes!

<table>
<thead>
<tr>
<th>SEQN</th>
<th>RIDAGEYR</th>
<th>BMXWAIST</th>
<th>BMXHT</th>
<th>LBXTC</th>
<th>BMXLEG</th>
<th>BMXWT</th>
<th>BMXBMI</th>
<th>RIDEETH1</th>
<th>BPQ2020</th>
<th>ALQ120Q</th>
<th>DMDEDUC2</th>
<th>RIAGENDR</th>
<th>INDFMPR</th>
<th>LBXGH</th>
<th>DIABETIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>73557</td>
<td>69.0</td>
<td>100.0</td>
<td>171.3</td>
<td>167.0</td>
<td>39.2</td>
<td>78.3</td>
<td>26.7</td>
<td>Non-Hispanic Black</td>
<td>yes</td>
<td>1.0</td>
<td>high school graduate / GED</td>
<td>male</td>
<td>0.84</td>
<td>13.9</td>
<td>yes</td>
</tr>
<tr>
<td>73558</td>
<td>54.0</td>
<td>107.6</td>
<td>178.8</td>
<td>170.0</td>
<td>40.0</td>
<td>89.5</td>
<td>28.6</td>
<td>Non-Hispanic White</td>
<td>yes</td>
<td>7.0</td>
<td>high school graduate / GED</td>
<td>male</td>
<td>1.78</td>
<td>9.1</td>
<td>yes</td>
</tr>
<tr>
<td>73559</td>
<td>72.0</td>
<td>109.2</td>
<td>175.3</td>
<td>126.0</td>
<td>40.0</td>
<td>88.9</td>
<td>28.9</td>
<td>Non-Hispanic White</td>
<td>yes</td>
<td>0.0</td>
<td>some college or AA degree</td>
<td>male</td>
<td>4.51</td>
<td>8.9</td>
<td>yes</td>
</tr>
<tr>
<td>73562</td>
<td>56.0</td>
<td>123.1</td>
<td>158.7</td>
<td>226.0</td>
<td>34.2</td>
<td>105.0</td>
<td>41.7</td>
<td>Mexican American</td>
<td>yes</td>
<td>5.0</td>
<td>some college or AA degree</td>
<td>male</td>
<td>4.79</td>
<td>5.5</td>
<td>no</td>
</tr>
<tr>
<td>73564</td>
<td>61.0</td>
<td>110.8</td>
<td>161.8</td>
<td>168.0</td>
<td>37.1</td>
<td>93.4</td>
<td>35.7</td>
<td>Non-Hispanic White</td>
<td>yes</td>
<td>2.0</td>
<td>college graduate or above</td>
<td>female</td>
<td>5.0</td>
<td>5.5</td>
<td>no</td>
</tr>
<tr>
<td>73566</td>
<td>56.0</td>
<td>85.5</td>
<td>152.8</td>
<td>278.0</td>
<td>32.4</td>
<td>61.8</td>
<td>26.5</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>1.0</td>
<td>high school graduate / GED</td>
<td>female</td>
<td>0.48</td>
<td>5.4</td>
<td>no</td>
</tr>
<tr>
<td>73567</td>
<td>65.0</td>
<td>93.7</td>
<td>172.4</td>
<td>173.0</td>
<td>40.0</td>
<td>65.3</td>
<td>22.0</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>4.0</td>
<td>9th-11th grade</td>
<td>male</td>
<td>1.2</td>
<td>5.2</td>
<td>no</td>
</tr>
<tr>
<td>73568</td>
<td>26.0</td>
<td>73.7</td>
<td>152.5</td>
<td>168.0</td>
<td>34.4</td>
<td>47.1</td>
<td>20.3</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>2.0</td>
<td>college graduate or above</td>
<td>female</td>
<td>5.0</td>
<td>5.2</td>
<td>no</td>
</tr>
<tr>
<td>73571</td>
<td>76.0</td>
<td>122.1</td>
<td>172.5</td>
<td>167.0</td>
<td>35.5</td>
<td>102.4</td>
<td>34.4</td>
<td>Non-Hispanic White</td>
<td>yes</td>
<td>2.0</td>
<td>college graduate or above</td>
<td>male</td>
<td>5.0</td>
<td>6.9</td>
<td>yes</td>
</tr>
<tr>
<td>73577</td>
<td>32.0</td>
<td>100.0</td>
<td>166.2</td>
<td>182.0</td>
<td>36.5</td>
<td>79.7</td>
<td>28.9</td>
<td>Mexican American</td>
<td>no</td>
<td>20.0</td>
<td>Less than 9th grade</td>
<td>male</td>
<td>0.29</td>
<td>5.3</td>
<td>no</td>
</tr>
<tr>
<td>73581</td>
<td>50.0</td>
<td>99.3</td>
<td>185.0</td>
<td>202.0</td>
<td>42.8</td>
<td>80.9</td>
<td>23.6</td>
<td>Other or Multi-Racial</td>
<td>no</td>
<td>0.0</td>
<td>college graduate or above</td>
<td>male</td>
<td>5.0</td>
<td>5.0</td>
<td>no</td>
</tr>
<tr>
<td>73585</td>
<td>28.0</td>
<td>90.3</td>
<td>175.1</td>
<td>198.0</td>
<td>40.5</td>
<td>92.2</td>
<td>30.1</td>
<td>Other or Multi-Racial</td>
<td>no</td>
<td>4.0</td>
<td>some college or AA degree</td>
<td>male</td>
<td>2.26</td>
<td>5.0</td>
<td>no</td>
</tr>
<tr>
<td>73589</td>
<td>35.0</td>
<td>94.6</td>
<td>172.9</td>
<td>192.0</td>
<td>39.1</td>
<td>78.3</td>
<td>26.2</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>2.0</td>
<td>high school graduate / GED</td>
<td>male</td>
<td>1.74</td>
<td>5.5</td>
<td>no</td>
</tr>
<tr>
<td>73595</td>
<td>58.0</td>
<td>114.8</td>
<td>175.3</td>
<td>165.0</td>
<td>40.1</td>
<td>96.0</td>
<td>31.2</td>
<td>Other Hispanic</td>
<td>no</td>
<td>1.0</td>
<td>some college or AA degree</td>
<td>male</td>
<td>3.09</td>
<td>7.7</td>
<td>no</td>
</tr>
<tr>
<td>73596</td>
<td>57.0</td>
<td>117.8</td>
<td>164.7</td>
<td>151.0</td>
<td>36.3</td>
<td>104.0</td>
<td>38.3</td>
<td>Other or Multi-Racial</td>
<td>yes</td>
<td>1.0</td>
<td>college graduate or above</td>
<td>female</td>
<td>5.0</td>
<td>5.9</td>
<td>no</td>
</tr>
<tr>
<td>73600</td>
<td>37.0</td>
<td>122.9</td>
<td>185.1</td>
<td>189.0</td>
<td>48.1</td>
<td>126.2</td>
<td>36.8</td>
<td>Non-Hispanic Black</td>
<td>yes</td>
<td>2.0</td>
<td>high school graduate / GED</td>
<td>male</td>
<td>0.63</td>
<td>6.2</td>
<td>yes</td>
</tr>
<tr>
<td>73604</td>
<td>69.0</td>
<td>96.6</td>
<td>156.9</td>
<td>203.0</td>
<td>37.0</td>
<td>59.5</td>
<td>24.2</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>1.0</td>
<td>some college or AA degree</td>
<td>female</td>
<td>2.44</td>
<td>5.4</td>
<td>no</td>
</tr>
<tr>
<td>73607</td>
<td>75.0</td>
<td>130.5</td>
<td>169.6</td>
<td>161.0</td>
<td>36.5</td>
<td>111.9</td>
<td>38.9</td>
<td>Non-Hispanic White</td>
<td>yes</td>
<td>0.0</td>
<td>high school graduate / GED</td>
<td>male</td>
<td>1.08</td>
<td>5.0</td>
<td>no</td>
</tr>
<tr>
<td>73610</td>
<td>43.0</td>
<td>102.6</td>
<td>176.8</td>
<td>200.0</td>
<td>38.8</td>
<td>90.2</td>
<td>28.9</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>5.0</td>
<td>college graduate or above</td>
<td>male</td>
<td>2.03</td>
<td>4.9</td>
<td>no</td>
</tr>
<tr>
<td>73613</td>
<td>60.0</td>
<td>113.6</td>
<td>163.8</td>
<td>203.0</td>
<td>41.6</td>
<td>104.9</td>
<td>39.1</td>
<td>Non-Hispanic Black</td>
<td>yes</td>
<td>2.0</td>
<td>9th-11th grade</td>
<td>female</td>
<td>5.0</td>
<td>6.1</td>
<td>no</td>
</tr>
<tr>
<td>73614</td>
<td>55.0</td>
<td>90.9</td>
<td>167.9</td>
<td>256.0</td>
<td>43.5</td>
<td>60.9</td>
<td>21.6</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>0.0</td>
<td>high school graduate / GED</td>
<td>female</td>
<td>1.29</td>
<td>5.0</td>
<td>no</td>
</tr>
<tr>
<td>73615</td>
<td>65.0</td>
<td>100.3</td>
<td>145.9</td>
<td>166.0</td>
<td>30.0</td>
<td>55.4</td>
<td>26.0</td>
<td>Other Hispanic</td>
<td>yes</td>
<td>1.0</td>
<td>Less than 9th grade</td>
<td>female</td>
<td>1.22</td>
<td>6.3</td>
<td>yes</td>
</tr>
<tr>
<td>73616</td>
<td>62.0</td>
<td>95.5</td>
<td>172.8</td>
<td>171.0</td>
<td>38.4</td>
<td>71.8</td>
<td>24.0</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>2.0</td>
<td>some college or AA degree</td>
<td>female</td>
<td>5.0</td>
<td>5.5</td>
<td>no</td>
</tr>
<tr>
<td>73619</td>
<td>36.0</td>
<td>91.1</td>
<td>173.1</td>
<td>162.0</td>
<td>38.9</td>
<td>81.7</td>
<td>27.3</td>
<td>Mexican American</td>
<td>no</td>
<td>2.0</td>
<td>high school graduate / GED</td>
<td>female</td>
<td>0.84</td>
<td>5.0</td>
<td>no</td>
</tr>
<tr>
<td>73621</td>
<td>80.0</td>
<td>98.2</td>
<td>176.2</td>
<td>161.0</td>
<td>40.4</td>
<td>76.4</td>
<td>24.6</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>5.0</td>
<td>college graduate or above</td>
<td>male</td>
<td>5.0</td>
<td>5.6</td>
<td>no</td>
</tr>
<tr>
<td>73622</td>
<td>72.0</td>
<td>115.6</td>
<td>185.4</td>
<td>186.0</td>
<td>39.7</td>
<td>99.5</td>
<td>28.9</td>
<td>Non-Hispanic White</td>
<td>no</td>
<td>4.0</td>
<td>college graduate or above</td>
<td>male</td>
<td>5.0</td>
<td>6.0</td>
<td>no</td>
</tr>
</tbody>
</table>

© 2019-22 D. Jayaraman, O. Bastani, Z. Ives
Recall the Basic Algorithm

function train_tree(\mathcal{D})

1. If data \mathcal{D} all have the same label y, return new leaf_node(y), else:

2. Pick the feature \( X_j \) to partition \mathcal{D} that maximizes Information Gain

3. Set node = new decision_node(\( X_j \))

4. For each value \( v \) that \( X_j \) can take
   Recursively create a new child train_tree(\( \mathcal{D}[X_j = v] \)) of node

5. Return node
## Entropy-Based Greedy DT Construction

Given dataset $D = [X, y]$
- **Pick feature $X_j$ to split upon with the highest IG (or GainRatio)**
- **Partition $D$ via $X_j$**
- **Recurse until nodes are homogenous ($0$ entropy)

### Dataset partition $D_{[LBXGH \leq 6.15]}$

<table>
<thead>
<tr>
<th>$X_1$</th>
<th>$X_2$</th>
<th>\ldots</th>
<th>$X_{14}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>False</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**GLYCOHEMOGLOBIN (LBXGH) $\leq 6.15$**
- entropy $= 0.533$
- samples $= 792$
- value $= [696, 96]$ 
- class $= None$

**GLYCOHEMOGLOBIN (LBXGH) $> 6.15$**
- entropy $= 0.412$
- samples $= 290$
- value $= [24, 256]$ 
- class $= Diabetes$

---

**X_{14} (LBXGH) $\leq 6.15$ has the highest IG**

---

**Dataset partition $D_{[LBXGH > 6.15]}$**

---

**Entropy**: $0.412$
- **Samples**: $290$
- **Value**: $[24, 256]$
- **Class**: $Diabetes$

---

**Entropy**: $0.533$
- **Samples**: $792$
- **Value**: $[696, 96]$
- **Class**: $None$
Diabetes DT – Random vs IG Features

DT with random feature splits

DT via IG

Accuracy on diabetes data = 100%
Accuracy on diabetes data = 100%

• It is smaller while retaining 100 % accuracy on our training data
• Still rather complex, though, and vulnerable to overfitting (we’ll see in a bit)...

• But first: let’s see a sketch of building Decision Trees in Scikit-Learn
An Example

Using Pandas, Numpy, Sklearn
# Classifying Mammals

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>bear</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dog</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>dolphin</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>bat</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>platypus</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>newt</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>skink</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>rat snake</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>lobster</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>kiwi</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>blue shark</td>
<td>N</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

*Inspired by an example by Mohsen Afsharchi*
Some Basics: Data into Pandas

animals_train_list = [
    {'organism': 'bear', 'endothermic': True, 'live_birth': True, 'four_legs': True, 'hibernates': True, 'fuzz': True, 'mammal': True},
    {'organism': 'dog', 'endothermic': True, 'live_birth': True, 'four_legs': True, 'hibernates': False, 'fuzz': True, 'mammal': True},
    {'organism': 'dolphin', 'endothermic': True, 'live_birth': True, 'four_legs': False, 'hibernates': False, 'fuzz': False, 'mammal': True},
    {'organism': 'bat', 'endothermic': True, 'live_birth': True, 'four_legs': False, 'hibernates': True, 'fuzz': True, 'mammal': True},
    {'organism': 'platypus', 'endothermic': True, 'live_birth': True, 'four_legs': True, 'hibernates': False, 'fuzz': True, 'mammal': True},
    {'organism': 'newt', 'endothermic': False, 'live_birth': True, 'four_legs': True, 'hibernates': True, 'fuzz': True, 'mammal': False},
    {'organism': 'skink', 'endothermic': False, 'live_birth': True, 'four_legs': True, 'hibernates': False, 'fuzz': True, 'mammal': False},
    {'organism': 'rat_snake', 'endothermic': False, 'live_birth': False, 'four_legs': False, 'hibernates': True, 'fuzz': False, 'mammal': False},
    {'organism': 'lobster', 'endothermic': False, 'live_birth': False, 'four_legs': False, 'hibernates': False, 'fuzz': False, 'mammal': False},
    {'organism': 'kiwi', 'endothermic': True, 'live_birth': False, 'four_legs': False, 'hibernates': False, 'fuzz': True, 'mammal': False},
    {'organism': 'blue_shark', 'endothermic': False, 'live_birth': True, 'four_legs': True, 'hibernates': True, 'fuzz': False, 'mammal': False}
]

animals_train_df = pd.DataFrame(animals_train_list)
## Some Basics: Identifying Features

<table>
<thead>
<tr>
<th>endothermic</th>
<th>live_birth</th>
<th>four_legs</th>
<th>hibernates</th>
<th>fuzz</th>
<th>mammal</th>
</tr>
</thead>
<tbody>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
</tr>
</tbody>
</table>

© 2019-22 D. Jayaraman, O. Bastani
Getting Training Data into Form

```python
X_train_df = animals_train_df[['endothermic', 'live_birth', 'four_legs', 'hibernates', 'fuzz']]
y_train_df = animals_train_df['mammal']

X_train = X_train_df.to_numpy()
y_train = y_train_df.to_numpy()

clf = tree.DecisionTreeClassifier()
trained = clf.fit(X_train,y_train)
tree.plot_tree(trained)
```
The Trained DT Model

X matrix columns

<table>
<thead>
<tr>
<th></th>
<th>Temp reg.?</th>
<th>Live birth?</th>
<th>Four legs?</th>
<th>Hibernates?</th>
<th>Fuzz/hair?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Temp Reg = False?

Live birth = False?

assuming nonmammal

assuming mammal
Overfitting and Decision Trees
Looks Perfect, Until We Test...

```python
X_test_df:

<table>
<thead>
<tr>
<th>organism</th>
<th>endothermic</th>
<th>live_birth</th>
<th>four_legs</th>
<th>hibernates</th>
<th>fuzz</th>
<th>mammal</th>
</tr>
</thead>
<tbody>
<tr>
<td>dolphin</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>human</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td>hairless_cat</td>
<td>True</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>whale</td>
<td>True</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>echidna</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>True</td>
</tr>
<tr>
<td>therizinosaurus</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td>True</td>
<td>False</td>
</tr>
<tr>
<td>bald_eagle</td>
<td>True</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
<td>False</td>
</tr>
</tbody>
</table>

trained.predict(X_test_df.to_numpy())

array([ True, True, True, True, False, False, False])
Similarly: with Diabetes Decision Tree

<table>
<thead>
<tr>
<th></th>
<th>Original Patient Data</th>
<th>New Patient Data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>100.00 %</td>
<td>82.796 %</td>
</tr>
<tr>
<td>(n = 1082)</td>
<td>(n = 465)</td>
<td></td>
</tr>
</tbody>
</table>
What is happening?

- Our algorithm has chosen some model representing hypothesis $h$
- But there (likely) exists another hypothesis $h'$ such that:

\[
\text{error}(h(D_{\text{train}})) < \text{error}(h'(D_{\text{train}}))
\]

\[
\text{error}(h'(D)) < \text{error}(h(D))
\]

(or else our heuristics are in fact preventing us from finding $h'$: our greedy algorithm doesn’t consider all possible trees)
What Causes Overfitting?

• Noisy training data: noise/errors can cause *contradictory labels* for data with the same features

• Training data is non-representative, or does not include unusual cases (e.g., egg-laying mammals, non-endothermic mammals)
Avoiding Overfitting

How can we avoid overfitting?

1. Acquire more training data (might be very hard)
2. Remove irrelevant attributes (manual process, not always possible)
3. Keep our model from getting too complex
Recall Occam’s Razor

Key Idea: The simplest consistent explanation is the best

What this Entails

• Have a *less complex* decision tree!

• This might actually look worse on training data but may generalize better!
Avoiding Overfitting

How can we avoid overfitting?

1. Acquire more training data
2. Remove irrelevant attributes (manual process, not always possible)
3. **Stop growing, e.g., when data split is not statistically significant**
4. **Grow full tree, then post-prune**

Try various tree hyperparameters (e.g., tree depth, splitting criterion, termination criterion) and pick the one with the **best estimated generalization performance**. How to estimate?

- Cross-validation
- Add a complexity penalty to performance measure e.g., training accuracy – average depth of leaf node
Stopping Growth

- Set a maximum **depth** to the decision tree (max_depth in Scikit-Learn)
- Set a minimum number of samples in a node, for us to split (e.g., 2) (min_samples_split in skl)
- Set a minimum number of samples in a leaf (min_samples_leaf)

(Again: we might use k-fold cross-validation to compare)

But alternatively, we can build “the perfect tree” and then **prune back**, based on validation set
Reduced-Error Pruning

Split the original training data into training and **validation sets**

**Training Stage**
Grow the decision tree based on the training set

**Pruning Stage**
Loop until further pruning hurts validation performance:
- Measure the validation performance of pruning each node (and its children)
- Greedily remove the node that most improves validation performance

This is very helpful for our Diabetes data
Reduced-Error Pruning

- Pruning replaces a **whole subtree with a leaf node**
- Replacement occurs if the error rate of the subtree is greater than that of the leaf

### Training

- **SpO2**
  - normal
  - low

### Validation

- **SpO2**
  - normal
  - low

**original subtree**

- (error rate = 4/6)

**pruned subtree**

- (error rate = 2/6)

Predicting the majority class (negative) has a lower validation error

Subtree should be pruned
Accuracy – Decision Trees

<table>
<thead>
<tr>
<th></th>
<th>DT unpruned</th>
<th>DT pruned</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original Patient Data:</td>
<td>100.000 %</td>
<td>88.909 %</td>
</tr>
<tr>
<td>New Patient Data:</td>
<td>82.796 %</td>
<td>85.591 %</td>
</tr>
</tbody>
</table>

(n = 1082)

(n = 465)
The Final Diabetes DT

Our Pruned Decision Tree

How Diabetes is Actually Diagnosed

- If your A1C level is between 5.7 and less than 6.5%, your levels have been in the prediabetes range.
- If you have an A1C level of 6.5% or higher, your levels were in the diabetes range.

(screenshot from diabetes.org)

Strong similarity to how diabetes is actually diagnosed!
Decision Tree Algorithms

**ID3**
- Information gain on nominal features

**C4.5**
- Can use info gain or gain ratio
- Nominal or numeric features
- Missing values
- Post-pruning
- Rule generation

**CART (Classification and Regression Tree)**
- Similar to C4.5
- Can handle continuous target prediction (regression)
- No rule sets
- Sklearn’s `DecisionTreeClassifier` is based on CART, but can’t handle nominal features (as of version 0.22.1)

**Many Other Algorithms …**
Strengths and Weaknesses of DTs

**Strengths**

👍 Widely used in practice
👍 Fast and simple to implement
👍 Small trees are easily interpretable
👍 Handles a variety of feature types
👍 Can convert to rules
👍 Handles noisy / missing data
👍 Insensitive to feature scaling
👍 Handles irrelevant features
👍 Handles large datasets

**Weaknesses**

👎 Univariate partitions limit potential trees
👎 Heuristic-Based greedy training
Another Idea to Prevent Overfitting

A single decision tree can be prone to overfitting to the training data.

What if we use randomization to create multiple decision trees, each a bit different:
- Each is trained on a sample of the training data
- Each splits along a subset of the possible features
- Each is a small decision tree (“stump”)

Then we rely on voting to make this work!
- Intuition: the most predictive features will be selected in many decision stumps!

(Note we now give up the “explainability” property)
Random Forests

(X_train, y_train): training data

subsets (with replacement) of (X_train, y_train)

separately trained decision tree classifiers using subsets of features

classifier combines votes of ensemble members
Training a Decision Tree in a Random Forest

1. Draw a random bootstrap data sample of size \( n \), with replacement

2. Build ("grow") a small decision tree (often just a "stump")
   - At each split point node, randomly select \( d \) candidate features (w/o replacement)
   - Split the node using the feature with best split according to objective function (e.g. information gain)

3. Repeat to produce \( k \) decision trees (a forest!)

4. For prediction, use **majority vote** to predict a class for new data
Benefits of Random Forests

One of the most popular and accurate classifiers for big data (more in a moment)

- Scale-invariant
- Much less susceptible to overfitting than “plain” decision trees
- Can be generalized to continuous data (random forests of CaRT trees)

Also: training is highly parallelizable!

- Take a data set, draw samples of size $n$ with replacement
- Train a separate decision tree on this, at each split point selecting from a subset of the features (without replacement for this tree)

Let’s see a case study...
Case Study: Seizure Prediction on Kaggle

American Epilepsy Society Seizure Prediction Challenge
Predict seizures in intracranial EEG recordings

Figure 1A
This device receives data from the implantable leads, predicts seizure activity using an algorithm, and sends an alert to a person or advisory device (cell phone, pager, etc.).

Figure 1B
Implantable Leads Assembly (ILA) Detects and relays electrical activity in the brain to the ITU.

Figure 1C
Single warning, repeated warnings triggered (no seizure occur), onset of full seizure.

Litt, Wagenaar et al. 2014
Multi-Channel Time Series Data
The Data

Data was broken into small fixed-length segments (labeled by a human expert)

Essentially just a 2D matrix: voltage levels per channel vs time

Kaggle divided data into training, leaderboard, and actual test sets

<table>
<thead>
<tr>
<th>#</th>
<th>Team</th>
<th>Members</th>
<th>Score</th>
<th>Entries</th>
<th>Last</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Medrr</td>
<td></td>
<td>0.83993</td>
<td>264</td>
<td>8Y</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>QMSDP</td>
<td></td>
<td>0.81962</td>
<td>501</td>
<td>8Y</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Birchwood</td>
<td></td>
<td>0.80078</td>
<td>160</td>
<td>8Y</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>ESI CEU-UCH</td>
<td></td>
<td>0.79347</td>
<td>182</td>
<td>8Y</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Data characteristics for the Kaggle.com seizure forecasting contest and held-out data experiment

<table>
<thead>
<tr>
<th>Subject</th>
<th>Sampling rate (Hz)</th>
<th>Recorded data (h)</th>
<th>Seizures Lead seizures</th>
<th>Training clips (%) interictal</th>
<th>Testing clips (%) interictal</th>
<th>Held-out clips (%) interictal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dog 1</td>
<td>400</td>
<td>1920</td>
<td>22</td>
<td>8</td>
<td>504 (95.2)</td>
<td>502 (95.2)</td>
</tr>
<tr>
<td>Dog 2</td>
<td>400</td>
<td>8208</td>
<td>47</td>
<td>40</td>
<td>54 (92.3)</td>
<td>1000 (91.0)</td>
</tr>
<tr>
<td>Dog 3</td>
<td>400</td>
<td>5112</td>
<td>104</td>
<td>18</td>
<td>1512 (95.2)</td>
<td>907 (95.4)</td>
</tr>
<tr>
<td>Dog 4</td>
<td>400</td>
<td>7152</td>
<td>29</td>
<td>27</td>
<td>901 (89.2)</td>
<td>990 (94.2)</td>
</tr>
</tbody>
</table>
• Just using the data samples wasn’t really enough!

• A few examples of techniques for extracting features from EEG timeseries:
  • Spectral analysis: use Fourier transforms to re-express the signal as a composition of sine waves, identify the frequency bands with the most power
  • look at the area under the curve or the derivative of the curve
  • etc.
Crowdsourcing reproducible seizure forecasting in human and canine epilepsy

Benjamin H. Brinkmann, Joost Wagenaar, Drew Abbot, Phillip Adkins, Simone C. Bosshard, Min Chen, Quang M. Tieng, Jialun He, F. J. Muñoz-Almaraz, Paloma Botella-Rocamora, Juan Pardo, Francisco Zamora-Martinez, Michael Hills, Wei Wu, Iryna Korshunova, Will Cukierski, Charles Vite, Edward E. Patterson, Brian Litt, Gregory A. Worrell

Author Notes

Published: 31 March 2016

Table 3 AUC scores for the held-out data experiment compared to scores on the public and private leader boards

<table>
<thead>
<tr>
<th>Team name</th>
<th>Window (overlap)</th>
<th>Features</th>
<th>Machine learning algorithm</th>
<th>Ensemble method</th>
<th>Public leader board</th>
<th>Private leader board</th>
</tr>
</thead>
<tbody>
<tr>
<td>QMSDP</td>
<td>60 s (0%)</td>
<td>Spectral power, spectral entropy, correlation, fractal dimensions</td>
<td>LassoGLM, Bagged SVM</td>
<td>Random Forest</td>
<td>0.86</td>
<td>0.82</td>
</tr>
<tr>
<td>QMSDP</td>
<td>8 s (97%)</td>
<td>Spectral power, correlation, signal variance</td>
<td>LassoGLM</td>
<td></td>
<td>0.84</td>
<td>0.81</td>
</tr>
<tr>
<td>QMSDP</td>
<td>8 s (97%)</td>
<td>Spectral power, correlation, signal variance</td>
<td>Bagged SVM</td>
<td></td>
<td>0.79</td>
<td>0.76</td>
</tr>
<tr>
<td>QMSDP</td>
<td>8 s (97%)</td>
<td>Spectral power, correlation, signal variance</td>
<td>Random Forest</td>
<td></td>
<td>0.79</td>
<td>0.72</td>
</tr>
</tbody>
</table>
Two Notes Here

- We saw from the competition that Random Forests – which used randomization to reduce overfitting (variance) in decision trees – were useful

- But additionally they combined many other kinds of classifiers

Are there some basic principles here?

*Ensembles*, next...