Creating Decision Trees for GLAST analysis: A new C++-based procedure

Toby Burnett
Frank Golf
Outline

- Quick review of classification (or decision) trees
- Training and testing
- How Bill does it with Insightful Miner
- Application to the “good-gamma” trees: how does it compare?
Quick Review of Decision Trees

- Introduced to GLAST by Bill Atwood, using InsightfulMiner
- Each branch **node** is a predicate, or cut on a variable, like \( CalCsIRLn > 4.222 \)
  - If true, this defines the **right** branch, otherwise the left branch.
  - If there is no branch, the node is a **leaf**; a leaf contains the **purity** of the sample that reaches that point
- Thus the tree defines a function of the event variables used, returning a value for the purity from the training sample
Training and testing procedure

- Analyze a training sample containing a mixture of “good” and “bad” events: I use the even events
- Choose set of variables and find the optimal cut for such that the left and right subsets are purer than the original. Two standard criteria for this: “Gini” and entropy. I currently use the former.
  - $W_S$: sum of signal weights
  - $W_B$: sum of background weights
  - $Gini = 2 \frac{W_S \cdot W_B}{(W_S + W_B)}$
    - Thus Gini wants to be small.
    - Actually maximize the improvement:
      $Gini(parent) - Gini(left\ child) - Gini(right\ child)$
- Apply this recursively until too few events. (100 for now)
- Finally test with the odd events: measure purity for each node
Evaluate by Comparing with Bill

From Bill’s Rome ’03 talk:

The “good cal” analysis

**CAL-Low CT Probabilities**

**CAL-High CT Probabilities**
Compare with Bill, cont

- Since Rome:
  - Three energy ranges, three trees.
    - CalEnergySum: 5-350; 350-3500; >3500
  - Resulting classification trees implemented in Gleam by a “classification” package: results saved to the IMgoodCal tuple variable.

- Data set for training, comparison: the all_gamma from v4r2
  - 760 K events E from 16 MeV to 160 GeV (uniform in log(E), and 0< θ < 90 (uniform in cos(θ)).
  - Contains IMgoodCal for comparison
Bill-type plots for all_gamma
Another way of looking at performance

Define efficiency as fraction of good events after given cut on node purity; determine bad fraction for that cut.
The new classifier package

- Properties
  - Implements Gini separation (entropy on todo list)
  - Reads ROOT files
  - Flexible specification of variables to use
  - Simple and compact ascii representation of decision trees
  - Not (yet) implemented: multiple trees, pruning

- Currently in users/burnett, I plan, after suitable notification, to copy it to the cvs root.
Growing trees

- For simplicity, run a single tree: initially try all of Bill’s classification tree variables: list at right.

- The run over the full 750 K events, trying each of the 70 variables takes only a few minutes!

```
"AcdActiveDist", "AcdTotalEnergy", "CalBkHalfRatio",
"CalCslRLn", "CalDeadTotRat", "CalDeltaT",
"CalEnergySum", "CalLATEdge", "CalLRmsRatio",
"CalLyr0Ratio", "CalLyr7Ratio", "CalMIPDiff",
"CalTotRLn", "CalTotSumCorr", "CalTrackDoaca",
"CalTrackSep", "CalTwrEdge", "CalTwrGap",
"CalXtalRatio", "CalXtalsTrunc", "EvtCalETLRatio",
"EvtCalETrackDoaca", "EvtCalETrackSep",
"EvtCalEXtalRatio", "EvtCalEXtalTrunc",
"EvtTkr1EChisq", "EvtTkr1EFirstChisq", "EvtTkr1EFrac",
"EvtTkr1EQual", "EvtTkr1PSFMrdRat",
"EvtTkr2EChisq", "EvtTkr2EFirstChisq",
"EvtTkrComptonRatio", "EvtTkrEComptonRatio",
"EvtTkrEdgeAngle", "EvtVtxEAngle", "EvtVtxEDoca",
"EvtVtxEEAngle", "EvtVtxEHeadSep",
"Tkr1Chisq", "Tkr1DieEdge", "Tkr1FirstChisq",
"Tkr1FirstLayer", "Tkr1Hits",
"Tkr1KalEne", "Tkr1PrjTwrEdge", "Tkr1Qual",
"Tkr1TwrEdge", "Tkr1ZDir", "Tkr2Chisq", "Tkr2KalEne",
"TkrBlankHits", "TkrHDCount", "TkrNumTracks",
"TkrRadLength", "TkrSumKalEne", "TkrThickHits",
"TkrThinHits", "TkrTotalHits", "TkrTrackLength",
"TkrTwrEdge", "VtxAngle", "VtxDOCA", "VtxHeadSep",
"VtxS1", "VtxTotalWgt", "VtxZDir"
```
Performance of the truncated list

1691 leaf nodes
3381 total

<table>
<thead>
<tr>
<th>Name</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalCsIRLn</td>
<td>21000</td>
</tr>
<tr>
<td>CalTrackDoca</td>
<td>3000</td>
</tr>
<tr>
<td>AcdTotalEnergy</td>
<td>1800</td>
</tr>
<tr>
<td>CalTotSumCorr</td>
<td>800</td>
</tr>
<tr>
<td>EvtCalEXtalTrunc</td>
<td>600</td>
</tr>
<tr>
<td>EvtTkr1EFrac</td>
<td>560</td>
</tr>
<tr>
<td>CalLyr7Ratio</td>
<td>470</td>
</tr>
<tr>
<td>CalTotRLn</td>
<td>400</td>
</tr>
<tr>
<td>CalEnergySum</td>
<td>370</td>
</tr>
<tr>
<td>EvtLogESum</td>
<td>370</td>
</tr>
<tr>
<td>EvtEnergySumOpt</td>
<td>310</td>
</tr>
<tr>
<td>EvtTkrEComptonRatio</td>
<td>290</td>
</tr>
<tr>
<td>EvtCalETrackDoca</td>
<td>220</td>
</tr>
<tr>
<td>EvtVtxEEAngle</td>
<td>140</td>
</tr>
</tbody>
</table>

Note: this is cheating: evaluation using the same events as for training.
Plans

- Implement multiple trees:
  - “boosting”
  - Averaging
- Try \textit{truncation} to reduce size of tree without losing performance
- This is only one of Bill’s variables: also need trees for:
  - PSF tail reduction
    - But we do not implement “regression trees”
  - vertex choice: one track or vertex
  - good-gamma prediction