1. Background
2. Hyper-parameter Optimization Methods
3. Distributed Computing
4. Experimental Results
Transformation of relevant data to high-quality descriptive and predictive models

Ex.: Neural Network
• How to determine model configuration?
• How to determine weights, activation functions?
BACKGROUND  MODEL TRAINING – OPTIMIZATION PROBLEM

• Objective Function

\[ f(w) = \frac{1}{n} \sum_{i=1}^{n} L(w; x_i, y_i) + \lambda_1 \|w\|_1 + \frac{\lambda_2}{2} \|w\|_2^2 \]

• Loss, \( L(w; x_i, y_i) \), for observation \( (x_i, y_i) \) and weights, \( w \)

• Stochastic Gradient Descent (SGD)

• Variation of Gradient Descent: \( w_{k+1} = w_k - \eta \nabla f(w_k) \)

• Approximate \( \nabla f(w_k) \) with gradient of mini-batch sample: \( \{x_{ki}, y_{ki}\} \)

\[ \nabla f_t(w) = \frac{1}{m} \sum_{i=1}^{m} \nabla L(w; x_{ki}, y_{ki}) \]
BACKGROUND

MODEL TRAINING – OPTIMIZATION PARAMETERS (SGD)

• Learning rate $\eta$
  • Too high, diverges
  • Too low, slow performance

• Momentum $\mu$
  • Too high, could “pass” solution
  • Too low, no performance improvement

• Regularization parameters $\lambda_1$ and $\lambda_2$
  • Too low, has little effect
  • Too high, drives iterates to 0

• Other parameters
  • Mini-batch size
  • Adaptive decay rate
  • Annealing rate
  • Communication frequency, ……
• Quality of trained model governed by so-called ‘hyper-parameters’
  No clear defaults agreeable to a wide range of applications

• Optimization options (SGD)

• Neural Network training options
  • Number of hidden layers
  • Number of neurons in each hidden layer
  • Random distribution for initial connection weights (normal, uniform, Cauchy)
  • Error function (gamma, normal, Poisson, entropy)
METHODS  HOW TO FIND GOOD HYPER-PARAMETER SETTINGS?

• Traditional Approach: *manual tuning*
  Even with expertise in machine learning algorithms and their parameters, best settings are directly dependent on the data used in training and scoring

• Hyper-parameter Optimization: *Grid vs. Random vs. “Real” Optimization*

  ![Standard Grid Search](image1)
  ![Random Search](image2)
  ![Random Latin Hypercube](image3)
METHODS  MANY CHALLENGES

Tuning objective, $T(x)$, is validation error score (to avoid increased overfitting effect)

Categorical / Integer Variables

Objective blows up

Flat-regions.

Node failure

Noisy/Nondeterministic c

Copyright © 2016, SAS Institute Inc. All rights reserved.
**METHODS**

**OUR APPROACH – LOCAL SEARCH OPTIMIZATION (LSO)**

Default hybrid search strategy:

1. **Initial Search: Latin Hypercube Sampling (LHS)**
2. **Global search: Genetic Algorithm (GA)**
   - Supports integer, categorical variables
   - Handles nonsmooth, discontinuous space
3. **Local search: Generating Set Search (GSS)**
   - Similar to Pattern Search
   - First-order convergence properties
   - Developed for continuous variables

All can be parallelized naturally
EXPERIMENTAL RESULTS

EXPERIMENT SETTINGS

• Decision tree:
  • Depth
  • Number of bins for interval variables
  • Splitting criterion

• Random Forest:
  • Number of trees
  • Number of variables at each split
  • Fraction of observations used to build a tree

• Gradient Boosting:
  • Number of trees
  • Number of variables at each split
  • Fraction of observations used to build a tree
  • L1 regularization
  • L2 regularization
  • Learning rate

• Neural Networks:
  • Number of hidden layers
  • Number of neurons in each hidden layer
  • LBFGS optimization parameters
  • SGD optimization parameters

SAS® Viya™

• Viya Data Mining and Machine Learning
• Small to medium sized datasets
EXPERIMENTAL RESULTS

EFFECTIVENESS OF THE DIFFERENT TUNING METHODS

- Gradient Boosting
  - 6 hyper-parameters
- 15 test problems
- Single 30% validation partition
- Single machine mode
- Run 10 times each
  - LSO (50x5 evaluations)
  - LHS (246 samples)
  - Random (246 samples)
- Averaged results

Average Improvement with Tuning
(Error reduction / Accuracy Increase)

Higher is better

<table>
<thead>
<tr>
<th></th>
<th>( \mu )</th>
<th>( \sigma )</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSO</td>
<td>10.661</td>
<td>2.193</td>
</tr>
<tr>
<td>LHS</td>
<td>11.085</td>
<td>2.324</td>
</tr>
<tr>
<td>Random</td>
<td>11.102</td>
<td>2.319</td>
</tr>
</tbody>
</table>
## EXPERIMENTAL RESULTS

### IMPACT ON THE DIFFERENT MACHINE LEARNING MODELS

- 13 test problems
- Single 30% validation partition
- Single machine mode
- Conservative default tuning process:
  - 5 Iterations
  - 10 configurations per iteration
- Run 10 times each
- Averaged results

### Average Improvement After Tuning

<table>
<thead>
<tr>
<th>ML</th>
<th>Average Error % Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>DT</td>
<td>5.2</td>
</tr>
<tr>
<td>GBT</td>
<td>5.0</td>
</tr>
<tr>
<td>RF</td>
<td>4.4</td>
</tr>
<tr>
<td>DT-P</td>
<td>3.0</td>
</tr>
</tbody>
</table>

Higher is better
EXPERIMENTAL RESULTS

MODELS – TIME VS. ACCURACY

- 13 test problems
- Single 30% validation partition
- Single Machine mode
- Conservative default tuning process:
  - 5 Iterations
  - 10 configurations per iteration
- Run 10 times each
- Averaged results

<table>
<thead>
<tr>
<th>ML</th>
<th>Average % Error</th>
<th>Average Time (Sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DT</td>
<td>15.3</td>
<td>9.2</td>
</tr>
<tr>
<td>DT-P</td>
<td>13.9</td>
<td>12.8</td>
</tr>
<tr>
<td>RF</td>
<td>12.7</td>
<td>49.1</td>
</tr>
<tr>
<td>GBT</td>
<td>11.7</td>
<td>121.8</td>
</tr>
</tbody>
</table>
EXPERIMENTAL RESULTS

SINGLE PARTITION VS. CROSS VALIDATION

- For each problem:
  - Tune with single 30% partition
  - Score best model on test set
  - Repeat 10 times
  - Average difference between validation error and test error
  - Repeat process with 5-fold cross validation

- Cross validation for small-medium data sets
  - 5x cost increase for sequential tuning process
  - manage in parallel / threaded environment
• Default train, 9.6% error
• Iteration 1 – Latin Hypercube best, 6.2% error

• Very similar configuration can have very different error
• Best after tuning: 2.6% error
EXPERIMENTAL RESULTS

PERFORMANCE ABNORMALITY OF PARALLEL TRAINING

IRIS Forest Tuning Time
105 Train / 45 Validate

Credit Data Tuning Time
49k Train / 21k Validate
LSO FOR HYPER-PARAMETER TUNING

HYBRID – PARALLEL TRAINING AND PARALLEL TUNING

- Hybrid tuning:
  - 4 nodes each training
  - n concurrent training
CONCLUSION  MANY TUNING OPPORTUNITIES AND CHALLENGES

- Initial Implementation
  - SAS® Viya™ Data Mining and Machine Learning
- Other search methods, extending hybrid solver framework
  - Bayesian / surrogate-based Optimization
  - New hybrid search strategies
- Selecting best machine learning algorithm
  - Parallel tuning across algorithms & strategies for effective node usage
  - Combining models
THANK YOU!

YAN.XU@SAS.COM