A Tailored Regression for Learned Indexes: Logarithmic Error Regression

Martin Eppert    Philipp Fent    Thomas Neumann

Technical University of Munich
Traditional Index Structures
Index Structures

Key

Model

Data
Learned Index Structures

Stage 3 Stage 2 Stage 1

Position

Key

Model 1.1

Model 2.1

Model 2.2

Model 2.3

... 

Model 3.1

Model 3.2

Model 3.3

Model 3.4

...
Learned Index Structures

Stage 1

Stage 2

Stage 3

Key

Position
Which Regression Model to Use?

Simple Linear Regression

Lasso
Robust
Theil-Sen
Non-Linear
Bayesian Linear
Least Absolute Deviation
Piecewise Linear
Least Squares
Polynomial
Logistic
Simple Linear Regression

- Optimizes Squared Error
- Linear runtime

\[ \hat{a} = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \]
\[ \hat{b} = \bar{y} - (\hat{a}\bar{x}) \]
Evaluation of Simple Linear Regression

• What metric do we want to optimize?

Lookup Time
Search Methods

Linear Search

Exponential Search
Search Methods

- Linear search
- Exponential search

Cost vs. Error graph showing the performance of linear and exponential search methods.
Search Methods

The graph illustrates the cost of different search methods as a function of error. The methods compared are linear search, squared error, and exponential search.

- **Linear Search**: The cost increases linearly with the error, as indicated by the blue line.
- **Squared Error**: The cost increases exponentially with error, as shown by the green dashed line.
- **Exponential Search**: The cost remains relatively constant for a range of error values, represented by the orange line.

The x-axis represents the error, while the y-axis represents the cost. The graph shows how each method performs under varying error conditions.
Logarithmic Error

\[ L(\epsilon) = \lceil \log_2(1 + \epsilon) \rceil \]
Is Optimization feasible?

\[ L(\epsilon) = \sum_{i=0}^{N} \log_2(1 + \epsilon) = \log_2 \left( \prod_{i=0}^{N} (1 + \epsilon) \right) \]
Is Optimization feasible?

![Graph showing Loss and Log Error as functions of b-Parameter]
Brute Force

• The regression must intersect 2 datapoints

• Test all combinations
  • $O(n^2)$ combinations
  • $O(n)$ time to calculate the error

• $O(n^3)$ time to calculate
Improving The Brute Force Algorithm

- \( \lceil \log_2 (1 + \epsilon) \rceil \) is a discrete function
- Takes on at most \( O(\log(\epsilon_{max})) \) discrete error values

- Idea: Find optimal regression intersecting a single datapoint
Improving The Brute Force Algorithm

Candidate0 Error = 4
Candidate1 Error = 3
Candidate2 Error = 2
Do we even need exact solutions?

- NO
- Least Squares Regression is already far off
- Not guaranteed to be optimal due to hardware specifics
Two Point Method

- Idea: create a gradient descent method finding appropriate local minima
- Obtain the optimal regression for a random datapoint
- Use the second crossed datapoint as the new pivot
- Repeat until convergence
Tournament Evaluation

- Idea: Not many samples are needed to determine a bad regression

- Pick $n$ candidates and let them compete in a tournament
- evaluate on $2^{height}$ datapoints
## MicroBenchmark

<table>
<thead>
<tr>
<th>Method</th>
<th>Data set</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tournament Evaluation Log</td>
<td>Facebook</td>
<td>6.9%</td>
<td>5.6%</td>
</tr>
<tr>
<td>Tournament Evaluation DLog</td>
<td>Facebook</td>
<td>6.7%</td>
<td>4.9%</td>
</tr>
<tr>
<td>Two Point Method</td>
<td></td>
<td>5.4%</td>
<td>4.3%</td>
</tr>
<tr>
<td>Tournament Evaluation Log</td>
<td>Wiki</td>
<td>3.9%</td>
<td>7.2%</td>
</tr>
<tr>
<td>Tournament Evaluation DLog</td>
<td>Wiki</td>
<td>3.9%</td>
<td>6.9%</td>
</tr>
<tr>
<td>Two Point Method</td>
<td></td>
<td>4.7%</td>
<td>7.0%</td>
</tr>
<tr>
<td>Tournament Evaluation Log</td>
<td>Normal</td>
<td>15.2%</td>
<td>14.7%</td>
</tr>
<tr>
<td>Tournament Evaluation DLog</td>
<td>Normal</td>
<td>15.1%</td>
<td>14.1%</td>
</tr>
<tr>
<td>Two Point Method</td>
<td></td>
<td>14.6%</td>
<td>13.9%</td>
</tr>
<tr>
<td>Tournament Evaluation Log</td>
<td>Outlier</td>
<td>121.6%</td>
<td>120.5%</td>
</tr>
<tr>
<td>Tournament Evaluation DLog</td>
<td>Outlier</td>
<td>120.5%</td>
<td>121.4%</td>
</tr>
<tr>
<td>Two Point Method</td>
<td></td>
<td>122.8%</td>
<td>121.9%</td>
</tr>
<tr>
<td>Tournament Evaluation Log</td>
<td>Uniform</td>
<td>1.6%</td>
<td>3.3%</td>
</tr>
<tr>
<td>Tournament Evaluation DLog</td>
<td>Uniform</td>
<td>1.4%</td>
<td>2.8%</td>
</tr>
<tr>
<td>Two Point Method</td>
<td></td>
<td>1.2%</td>
<td>3.7%</td>
</tr>
</tbody>
</table>
Robustness
Bounds Checking

![Graph showing Log Regression and Least Squares](image)

Log Regression

Least Squares
## Benchmark on the RMI

<table>
<thead>
<tr>
<th>Data set</th>
<th>1&lt;sup&gt;st&lt;/sup&gt; Layer</th>
<th>Lookup [ns]</th>
<th>Build [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SLR</td>
<td>Log</td>
<td>Speedup</td>
</tr>
<tr>
<td>Facebook</td>
<td>linear</td>
<td>245.7</td>
<td>244.1</td>
</tr>
<tr>
<td>Wiki</td>
<td>linear</td>
<td>172.1</td>
<td>160.6</td>
</tr>
<tr>
<td>Osm</td>
<td>cubic</td>
<td>399.4</td>
<td>383.4</td>
</tr>
<tr>
<td>Gaussian Mixture</td>
<td>linear</td>
<td>296.7</td>
<td>266.0</td>
</tr>
</tbody>
</table>
Lessons Learned

![Graph showing mean log error and mean squared error with least log error of 3.7 and least squared error of 2760.](image-url)
Conclusion

• Standard regression is suboptimal
• Proposed methods fit underlying search method
• Logarithmic error is an important measure
• Show improvement on the Recursive Model Index
Future Work

- Build time optimized regression
- Detect if the logarithmic error regression is needed
- Integrate cache lines into the error function