How Developers Iterate on Machine Learning Workflows

-- A Survey of the Applied Machine Learning Literature

Doris Xin\textsuperscript{1}, Litian Ma\textsuperscript{1}, Shuchen Song\textsuperscript{1}, Rong Ma\textsuperscript{2}, Aditya Parameswaran\textsuperscript{1}

\textsuperscript{1} University of Illinois at Urbana-Champaign

\textsuperscript{2} Peking University
Developing Machine Learning Applications is **Iterative**

1. **Input data**
2. **Data Preprocessing**
   - Data cleaning, feature eng., etc.
3. **Learning/Inference**
   - Train ML models, Make predictions w/ trained models
4. **Post Processing**
   - Evaluation, Interpretation & Explanation

- Add features, scale features, etc.
- Add regularization, change model type, etc.
- Change metrics, drill down on results, etc.
Developing Machine Learning Applications is Interactive!

Creating systems to enhance interactivity requires a statistical characterization of how developers iterate on ML workflows.
How Do Developers Iterate on Machine Learning Workflows?
How Do Developers Iterate on Machine Learning Workflows?

Our approach: study iterations by collecting statistics from applied ML papers grouped by application domains.
Outline

- Data & Limitations
- Methodology
  - Statistics
  - Estimation
- Results
- Conclusion & Future Work
Outline

● **Data & Limitations**
● Methodology
  ○ Statistics
  ○ Estimation
● Results
● Conclusion & Future Work
Corpus: 105 Papers from 2016

Limitations
- Incomplete picture of iterations
  - Focus on ML and omit DPR
- Results presented side-by-side
  - Can’t determine the order
- # papers / domain is small
  - May lead to spurious results

Remedies
- Multiple surveyors to reduce chance of spurious results
- Iteration estimators that do not rely on order
Outline

● Data & Limitations
● **Methodology**
  ○ Statistics
  ○ Estimation
● Results
● Conclusion & Future Work
Collecting Statistics

<table>
<thead>
<tr>
<th>Data Prep.</th>
<th>ML Model Class</th>
<th>ML Tuning</th>
<th>Evaluation Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>norm.</td>
<td>impute</td>
<td>LSTM</td>
<td>Reg.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SVM</td>
<td>Learn. Rate</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>AUC</td>
</tr>
</tbody>
</table>

Aggregate

Open source dataset at https://github.com/helix-ml/AppliedMLSurvey
Estimating Iterations

<table>
<thead>
<tr>
<th>Data Prep.</th>
<th>ML Model Class</th>
<th>ML Tuning</th>
<th>Evaluation Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>norm.</td>
<td>...</td>
<td>LSTM</td>
<td>Reg.</td>
</tr>
<tr>
<td>impute</td>
<td>...</td>
<td>SVM</td>
<td>Learn. Rate</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Aggregate

| Number of data prep. iterations | $t_{DPR}$ |
| Number of ML iterations        | $t_{LI}$  |
| Number of post proc. iterations| $t_{PPR}$ |

<table>
<thead>
<tr>
<th># tables</th>
<th># figs</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2</td>
</tr>
</tbody>
</table>
Outline

- Data & Limitations
- Methodology
  - Statistics
  - Estimation
- Results
- Conclusion & Future Work
Mean Iteration Count by Domains

- Social Sciences
- Natural Sciences
- Web App
- NLP
- CV

E[t_{DPR}]
E[t_{LI}]
E[t_{PPR}]

0 1 2 3 4 5 6 7
Data Preprocessing

<table>
<thead>
<tr>
<th>Social Sciences</th>
<th>Natural Sciences</th>
<th>Web Apps</th>
<th>NLP</th>
<th>Computer Vision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Join (31.0%)</td>
<td>Feat. Def. (40.6%)</td>
<td>Feat. Def. (36.1%)</td>
<td>Feat. Def. (32.1%)</td>
<td>Feat. Def. (37.5%)</td>
</tr>
<tr>
<td>Feat. Def. (27.6%)</td>
<td>Univar. FS (18.8%)</td>
<td>Join (22.2%)</td>
<td>BOW (17.9%)</td>
<td>BOW (25.0%)</td>
</tr>
<tr>
<td>Normalize (17.2%)</td>
<td>Normalize (12.5%)</td>
<td>Normalize (13.9%)</td>
<td>Join (14.3%)</td>
<td>Interaction (25.0%)</td>
</tr>
<tr>
<td>Impute (6.9%)</td>
<td>PCA (9.4%)</td>
<td>Discretize (8.3%)</td>
<td>Normalize (10.7%)</td>
<td>Join (12.5%)</td>
</tr>
</tbody>
</table>

- **Feat. Def.** = human defined features from raw attributes
  - e.g. adult=true if age >=18
### ML Model Classes

<table>
<thead>
<tr>
<th>Social Sciences</th>
<th>Natural Sciences</th>
<th>Web Apps</th>
<th>NLP</th>
<th>Computer Vision</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLM (36.0%)</td>
<td>SVM (32.7%)</td>
<td>GLM (37.0%)</td>
<td>RNN (32.4%)</td>
<td>CNN (38.2%)</td>
</tr>
<tr>
<td>SVM (28.0%)</td>
<td>GLM (15.4%)</td>
<td>SVM (11.1%)</td>
<td>GLM (14.7%)</td>
<td>SVM (17.6%)</td>
</tr>
<tr>
<td>RF (20.0%)</td>
<td>RF (13.5%)</td>
<td>RF (11.1%)</td>
<td>SVM (11.8%)</td>
<td>RNN (17.6%)</td>
</tr>
<tr>
<td>Decision Tree (12.0%)</td>
<td>DNN (13.5%)</td>
<td>Matrix Factor. (11.1%)</td>
<td>CNN (8.8%)</td>
<td>RF (5.9%)</td>
</tr>
</tbody>
</table>

- Generalized linear models: logistic regression, linear regressions, etc.
- **SVMs** are popular (especially in natural sciences!) possibly due to kernels
- **Deep learning** is only popular in NLP and computer vision so far
## ML Model Tuning

<table>
<thead>
<tr>
<th>Social Sciences</th>
<th>Natural Sciences</th>
<th>Web Apps</th>
<th>NLP</th>
<th>Computer Vision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regularize (40.0%)</td>
<td>Cross Val. (31.8%)</td>
<td>Regularize (41.2%)</td>
<td>Learn Rate (39.4%)</td>
<td>Learn Rate (46.2%)</td>
</tr>
<tr>
<td>Cross Val. (30.0%)</td>
<td>Learn Rate (22.7%)</td>
<td>Learn Rate (23.5%)</td>
<td>Batch Size (24.2%)</td>
<td>Batch Size (30.8%)</td>
</tr>
<tr>
<td>Learn Rate (10.0%)</td>
<td>DNN Arch. (18.2%)</td>
<td>Batch Size (11.8%)</td>
<td>DNN Arch. (18.2%)</td>
<td>DNN Arch. (11.5%)</td>
</tr>
<tr>
<td>Batch Size (10.0%)</td>
<td>Kernel (9.1%)</td>
<td>Cross Val. (11.8%)</td>
<td>Kernel (6.1%)</td>
<td>Regularize (11.5%)</td>
</tr>
</tbody>
</table>

- Learning Rate + Batch Size → looking for faster training
### Post Processing

<table>
<thead>
<tr>
<th>Social Sciences</th>
<th>Natural Sciences</th>
<th>Web Apps</th>
<th>NLP</th>
<th>Computer Vision</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Prec/Rec</strong> (25.7%)</td>
<td><strong>Accuracy</strong> (28.6%)</td>
<td><strong>Accuracy</strong> (20.8%)</td>
<td><strong>Prec/Rec</strong> (29.2%)</td>
<td><strong>Visualiz.</strong> (33.3%)</td>
</tr>
<tr>
<td><strong>Accuracy</strong> (20.0%)</td>
<td><strong>Prec/Rec</strong> (18.6%)</td>
<td><strong>Prec/Rec</strong> (20.8%)</td>
<td><strong>Accuracy</strong> (27.1%)</td>
<td><strong>Accuracy</strong> (29.8%)</td>
</tr>
<tr>
<td><strong>Feat. Contrib.</strong> (17.1%)</td>
<td><strong>Visualiz.</strong> (15.7%)</td>
<td><strong>Case Studies</strong> (13.2%)</td>
<td><strong>Case Studies</strong> (14.6%)</td>
<td><strong>Prec/Rec</strong> (17.5%)</td>
</tr>
<tr>
<td><strong>Visualiz.</strong> (14.3%)</td>
<td><strong>Correlation</strong> (11.4%)</td>
<td><strong>DCG</strong> (9.4%)</td>
<td><strong>Human Eval</strong> (8.3%)</td>
<td><strong>Case Studies</strong> (12.3%)</td>
</tr>
</tbody>
</table>

- Precision/Recall & Accuracy → coarse-grained evaluation
- Case Studies & Visualization → fine-grained evaluation
Takeaways

- Study iteration using **empirical evidence** from applied ML papers
  - Grouping by domains gives better insights
- Lessons from results
  - **Data prep**: fine-grained feature engineering, efficient joins
  - **ML**: explainable models and fast training
  - **Eval**: fine-grained evals are as common as coarse-grained metrics
- Open source dataset at [https://github.com/helix-ml/AppliedMLSurvey](https://github.com/helix-ml/AppliedMLSurvey)
Future Work

- Refine statistics and estimators
- Develop insights and trends into a benchmark
- Look at code repositories (e.g. Kaggle) for a more complete picture

Accelerate Iterative Execution via Intermediates Reuse

- Address user needs discovered in our survey
- Selectively materialize intermediate results for reuse in future iterations

https://helix-ml.github.io

More on Helix in the technical report @ http://data-people.cs.illinois.edu/helix-tr.pdf