Data-Intensive Distributed Computing
CS 431/631 451/651 (Winter 2019)

Part 6: Data Mining (2/4)
October 31, 2019

Ali Abedi

These slides are available at https://www.student.cs.uwaterloo.ca/~cs451

This work is licensed under a Creative Commons Attribution-Noncommercial-Share Alike 3.0 United States
See http://creativecommons.org/licenses/by-nc-sa/3.0/us/ for details
Stochastic Gradient Descent
Batch vs. Online

Gradient Descent

\[
\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \frac{1}{n} \sum_{i=0}^{n} \nabla \ell(f(x_i; \theta^{(t)}), y_i)
\]

“batch” learning: update model after considering all training instances

Stochastic Gradient Descent (SGD)

\[
\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \nabla \ell(f(x; \theta^{(t)}), y)
\]

“online” learning: update model after considering each (randomly-selected) training instance

In practice... just as good!
Opportunity to interleaving prediction and learning!
Practical Notes

Order of the instances important!
Most common implementation: randomly shuffle training instances

Single vs. multi-pass approaches

Mini-batching as a middle ground

We’ve solved the iteration problem!
What about the single reducer problem?
Ensembles
Ensemble Learning

*independent*

Learn multiple models, combine results from different models to make prediction.

Common implementation:
Train classifiers on different input partitions of the data
Embarrassingly parallel!

Combining predictions:
- Majority voting
- Model averaging
Ensemble Learning

Learn multiple models, combine results from different models to make prediction

Why does it work?
If errors uncorrelated, multiple classifiers being wrong is less likely
Reduces the variance component of error
MapReduce Implementation

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \nabla \ell(f(x; \theta^{(t)}), y)$$
MapReduce Implementation

\[ \theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \nabla \ell(f(x; \theta^{(t)}), y) \]
MapReduce Implementation

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \nabla \ell(f(x; \theta^{(t)}), y)$$

How do we output the model?

Option 1: write model out as “side data”
Option 2: emit model as intermediate output
What about Spark?

\[ \theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \nabla \ell(f(x; \theta^{(t)}), y) \]
Classifier Training

Making Predictions

Just like any other parallel Pig dataflow
Classifier Training

training = load ‘training.txt’ using SVMLightStorage()
as (target: int, features: map[]);

store training into ‘model/’
using FeaturesLRClassifierBuilder();

Logistic regression + SGD (L2 regularization)

Want an ensemble?

training = foreach training generate
    label, features, RANDOM() as random;
training = order training by random parallel 5;
Making Predictions

define Classify ClassifyWithLR('model/');
data = load 'test.txt' using SVMLightStorage() as (target: double, features: map[]);
data = foreach data generate target, Classify(features) as prediction;

Want an ensemble?

define Classify ClassifyWithEnsemble('model/', 'classifier.LR', 'vote');
Sentiment Analysis Case Study

Binary polarity classification: \{positive, negative\} sentiment

Use the “emoticon trick” to gather data

Data

Test: 500k positive/500k negative tweets from 9/1/2011
Training: \{1m, 10m, 100m\} instances from before (50/50 split)

Features:

Sliding window byte-4grams

Models + Optimization:

Logistic regression with SGD (L2 regularization)
Ensembles of various sizes (simple weighted voting)
Ensembles with 10m examples better than 100m single classifier!

For free
Supervised Machine Learning

Machine Learning Algorithm

training data

training

testing/deployment

Model

17
Evaluation

How do we know how well we’re doing?

Why isn’t this enough?

Induce: \( f : X \rightarrow Y \)

Such that loss is minimized

\[
\arg \min_{\theta} \frac{1}{n} \sum_{i=0}^{n} \ell(f(x_i; \theta), y_i)
\]

We need end-to-end metrics!

Obvious metric: accuracy

Why isn’t this enough?
Metrics

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predicted</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>Positive</td>
</tr>
<tr>
<td>Positive</td>
<td>False Negative (FN) = Type II Error</td>
</tr>
<tr>
<td>Negative</td>
<td>True Negative (TN)</td>
</tr>
</tbody>
</table>

- **True Positive (TP)**
- **False Negative (FN)**
- **False Positive (FP)**
- **True Negative (TN)**

**Precision**

\[
\text{Precision} = \frac{TP}{(TP + FP)}
\]

**Miss rate**

\[
\text{Miss rate} = \frac{FN}{(FN + TN)}
\]

**Recall or TPR**

\[
\text{Recall or TPR} = \frac{TP}{(TP + FN)}
\]

**Fall-Out or FPR**

\[
\text{Fall-Out or FPR} = \frac{FP}{(FP + TN)}
\]
ROB and PR Curves

Source: Davis and Goadrich. (2006) The Relationship Between Precision-Recall and ROC curves
Training/Testing Splits

Training

\[ \arg\min_{\theta} \frac{1}{n} \sum_{i=0}^{n} \ell(f(x_i; \theta), y_i) \]

Test

Precision, Recall, etc.

What happens if you need more?

Cross-Validation
Training/Testing Splits

Cross-Validation
Training/Testing Splits

Cross-Validation
Training/Testing Splits

Cross-Validation
Training/Testing Splits

Cross-Validation
Training/Testing Splits

Cross-Validation
Typical Industry Setup

Training

Test

A/B test

Why not cross-validation?
A/B Testing

Gather metrics, compare alternatives
A/B Testing: Complexities

Properly bucketing users

Novelty

Learning effects

Long vs. short term effects

Multiple, interacting tests

Nosy tech journalists

...
Supervised Machine Learning

- Training data
- Machine Learning Algorithm

Training | Testing/Deployment

Model
Applied ML in Academia

Download interesting dataset (comes with the problem)

Run baseline model
  Train/Test

Build better model
  Train/Test

Does new model beat baseline?
  Yes: publish a paper!
  No: try again!
Data Scientist: The Sexiest Job of the 21st Century

by Thomas H. Davenport and D.J. Patil

FROM THE OCTOBER 2012 ISSUE
<table>
<thead>
<tr>
<th>Fantasy</th>
<th>Reality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract features</td>
<td>What’s the task?</td>
</tr>
<tr>
<td>Develop cool ML technique</td>
<td>Where’s the data?</td>
</tr>
<tr>
<td>#Profit</td>
<td>What’s in this dataset?</td>
</tr>
<tr>
<td></td>
<td>What’s all the f#$!* crap?</td>
</tr>
<tr>
<td></td>
<td>Clean the data</td>
</tr>
<tr>
<td></td>
<td>Extract features</td>
</tr>
<tr>
<td></td>
<td>“Do” machine learning</td>
</tr>
<tr>
<td></td>
<td>Fail, iterate…</td>
</tr>
</tbody>
</table>

Dirty secret: very little of data science is about machine learning per se!
It’s impossible to overstress this: 80% of the work in any data project is in cleaning the data. – DJ Patil “Data Jujitsu”
For ‘Big Data’ Scientists, Hurdle to Insights Is ‘Janitor Work’

By STEVE LOHK  Aug. 11, 2014

Monica Rogati, Jawbone’s vice president for data science, with Brian Wilt, a senior data scientist.

Peter DaSilva for The New York Times
On finding things...

P. Oscar Boykin
@posco

OH: "... so to recap, tweets are statuses, favorites are favourings, retweets are shares."
On naming things...

CamelCase

smallCamelCase

snake_case

camel_Snake

dunder___snake

Yesterday I had a run in with the camel_Snake in our code. Today, I came across the feared dunder___snake. Yow! /via @THISWILLWORK

10:46 PM - Sep 12, 2012  from SoMa, San Francisco
On feature extraction...

An actual Java regular expression used to parse log message at Twitter circa 2010

Friction is cumulative!
[scene: consumer internet company in the Bay Area...]

Frontend Engineer
Develops new feature, adds logging code to capture clicks

Data Scientist
Analyze user behavior, extract insights to improve feature

Okay, let’s get going... where’s the click data?
Well, that’s kinda non-intuitive, but okay...

Oh, BTW, where’s the timestamp of the click?

Hang on, I don’t remember...

Uh, bad news. Looks like we forgot to log it...

[grumble, grumble, grumble]

Data Plumbing... Gone Wrong!
<table>
<thead>
<tr>
<th>Fantasy</th>
<th>Reality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extract features</td>
<td>What’s the task?</td>
</tr>
<tr>
<td>Develop cool ML technique</td>
<td>Where’s the data?</td>
</tr>
<tr>
<td>#Profit</td>
<td>What’s in this dataset?</td>
</tr>
<tr>
<td>Clean the data</td>
<td>What’s all the f#$!* crap?</td>
</tr>
<tr>
<td>Extract features</td>
<td>Clean the data</td>
</tr>
<tr>
<td>“Do” machine learning</td>
<td>Extract features</td>
</tr>
<tr>
<td>Fail, iterate...</td>
<td>“Do” machine learning</td>
</tr>
<tr>
<td></td>
<td>Fail, iterate...</td>
</tr>
</tbody>
</table>

Finally works!
Congratulations, you’re halfway there...
Congratulations, you’re halfway there...

Does it actually work?
A/B testing

Is it fast enough?

Good, you’re two thirds there...
Productionize
Productionize

What are your jobs’ dependencies?
How/when are your jobs scheduled?
    Are there enough resources?
    How do you know if it’s working?
Who do you call if it stops working?

Infrastructure is critical here!
(plumbing)
Takeaway lesson:
Most of data science isn’t glamorous!

Source: Wikipedia (Plumbing)