



Storage, Security, and Privacy in the age of ML

**Aleatha Parker-Wood, Ph.D.
Machine Learning and Privacy
Lead, Humu**



Who am I?

- Storage systems Ph.D. from UCSC
- Post-doc in a databases group
- Joined Symantec Research Labs....
 - Right before the Veritas split
 - Chose to stay on the security side
- Spent the last five years focused on ML and algorithmic privacy in security
- Now leading ML and algorithmic privacy at Humu (we make work better!)
- This talk will cover both sides of the storage/ML fence

Why are we talking about this?

- It's always worth pausing and asking whether something should be discussed
- AI/ML is a hot topic!
- Storage is, I'm assuming, what you all do (Unless I've wandered into the wrong room again.)
- Does that mean we should discuss ML and storage together?
- Is machine learning fundamentally different from many other storage problems?



Spoiler alert: it's not (always) different. And yet...

- ML is a really diverse field
- There's a mammoth range of algorithms and tools
- Good news (if you're a practitioner):
 - Most of these algorithms will leverage already existing storage and data structure paradigms
 - Trends in ML will likely drive a huge variety of system designs, but we have many of the right tools already
- Good news (if you're a researcher)
 - We still need to worry about scale and latency (and power)
 - We have some serious problems around security and privacy
 - GDPR and its siblings will reach into every aspect of system design

Some bitter truths

- The vast majority of ML in industry...
- ...Is not big data
- ...If it is big data, the bulk of the work is data pre-processing
- ...Is done on a single machine (often a laptop) or a tiny handful of machines
- ..Is not deep learning
- ...Does the job just fine

More bitter truths

- Storage is where the rubber hits the road, and bad infrastructure can ruin everyone's day
- ML has created huge “attractive nuisances” in the storage domain
- Never has so much valuable, vulnerable data been aggregated in one place
- Making privacy and security a first class design constraint is more important today than ever
- We can always solve it in middleware, but it will often be inefficient and slow and possibly insecure

Roadmap

Trends in ML (and what they mean
for storage and distributed systems)

Security and privacy are everyone's
problem



Trends in AI and their implications



A Quick intro to ML terminology and workflow

- Sample: A single data point (image/email/virus/song) and its properties.
- Feature: The properties of a sample. Pixels, words, file entropy, a Fourier transform.
- Labeled data: “This sample is a dog/spam/a virus/a Prince song”
- Unlabeled data: “I don’t know what this is.”
- Supervised training: “I will take data and labels and create a function that can predict new labels (a model)”
- Unsupervised training: “I will try to find a compact function to describe this data.”
- Inference: “I will take a new sample, run it through the model, and get a label.”

ML terminology in a nutshell

Sample →	Features				Label
		2018-10-07	USA	...	Dog
		2019-01-03	Belgium	...	Cat
		1992-05-05	USA	...	Gazebo
		2018-01-20	Australia	...	Macadamia
		1964-06-12	USA	...	Bulbous bouffant

Evergreen: Semistructured data, supervised learning

- Still the bread and butter of industry ML!
- Mostly used for: handcrafted features, hand-labeled data
- In many ways, basically a database workload...
- ...except we rarely know the important columns in advance of training
- We know how to handle this!

Tabular data and storage workflow

- Much of the “Big Data” stack was born here
- Map-reduce, Spark, Hadoop
- Mixed and compute storage nodes
- Data is shuffled and distributed to the storage nodes
- Map functions run in parallel over the data
- Reduce functions crunch it down to model parameters

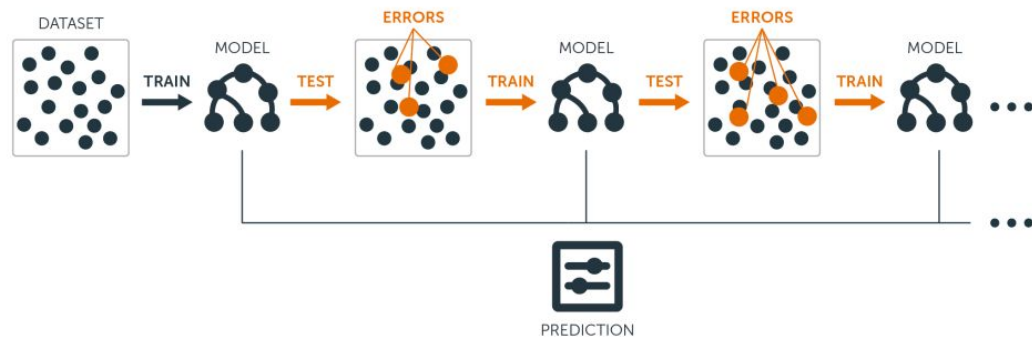
Some models are easier than others

- Easy: Random forests

- Select a random subset of columns and rows
- Train a decision tree on it
- Return the tree
- Eminently parallel

- Harder: Boosted Random Forests

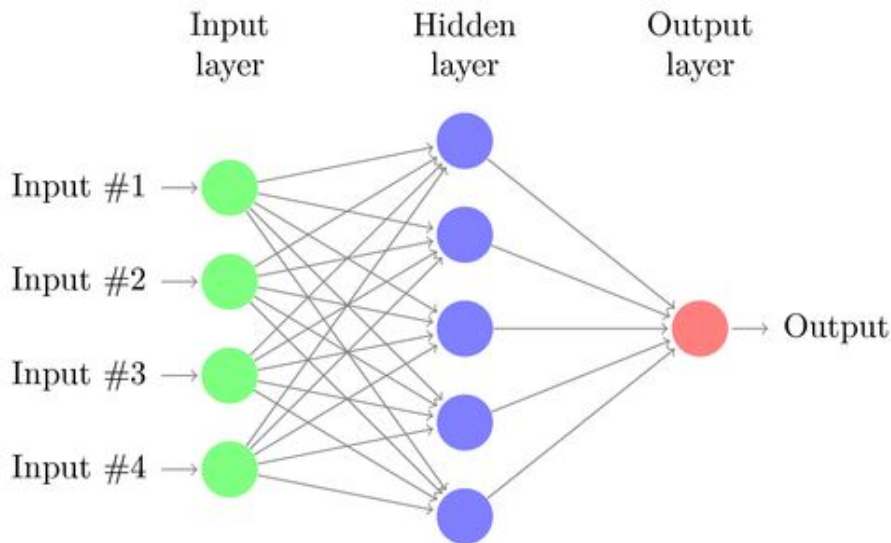
- The workhorse of industrial ML
- When you make a mistake in your decision tree, save it, then train more trees on your failures
- Minor algorithmic change
- ruins parallelism!



What about deep learning?!

- Currently a hot trend in ML (but not the only one)
- Commonly used for: Images, NLP, physics
 - Physics is not my area of expertise, but fortunately Quincey is talking soon.
- Blobs, tabular data, or sequences
- Supervised or unsupervised
- Insanely data and power-hungry. (Recent estimates are on order of 1 American carbon-year for a state-of-the-art NLP model with tuning.)

<http://www.texample.net/tikz/examples/neural-network/>



Deep learning and storage

- Parameter server
 - Maintains the neural network parameters (a series of large matrices)
 - Frequent small updates
 - Hopefully all in memory!
- Workers and mini-batches
 - Each worker pulls subsets of samples from storage (a “minibatch”), order 10-100 samples
 - Processes a few at a time (Usually limited by GPU memory)
 - Send updates to parameter server
 - Shuffle all samples and redistribute
 - An “epoch” is a single pass over all the data
 - Models often take many epochs to converge (more data == fewer epochs, but 500-4K is not uncommon)

Trending: Causal Inference

- ML is great at drawing out correlations
- But not causes
- Spurious correlations lead to bad models
- For example:
 - If there is a person and a phone in the same image, it'll be labeled as "phone call", even if the person is nowhere near the phone. (Oquab et al., CVPR 2014)
- One solution: maintain context to find invariants! Time and place should stay together.
- Less shuffling, more sequential reads (cool.)
- Needs a lot more data to train. (Less cool.)

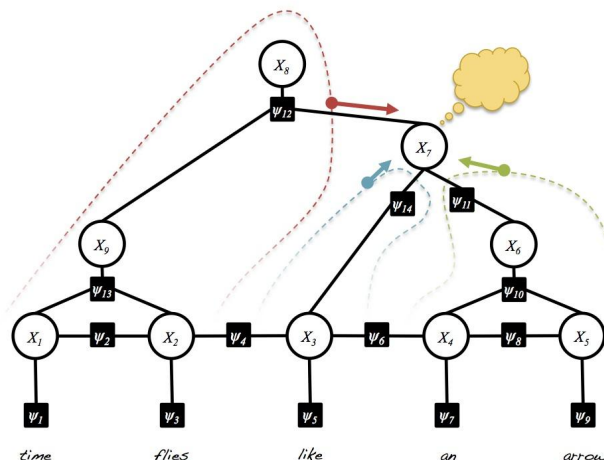


Trending: Graph algorithms

- What are the statistical relationships between many things? (E.g. probabilistic graphical models, PageRank, belief propagation)
- Used for: modeling complex interactions between many variables
- Requires efficient representation and retrieval of graphs
- Poor access locality
- Semi-structured data
- Lots of message passing
- Really rough on storage systems when done at scale

Loopy Belief Propagation

What if our graph has cycles?



- Messages from different subgraphs are **no longer independent!**
 - Dynamic programming can't help. It's now #P-hard in general to compute the exact marginals.
- **But we can still run BP** -- it's a local algorithm so it doesn't "see the cycles."



**Security and Privacy are
first class storage
concerns**



Now let's talk about the important stuff

- Machine learning has been driven by a wide variety of applications
- Ad placement. Facial recognition. Mapping. Self-driving cars.
- We need to acknowledge that the vast majority of data stored for ML is about people, created by people, and has the potential for harm.
- Storage plays a huge role in making sure that data is secure by default and treated with care and respect throughout its lifetime

Fun GDPR fact!

- GDPR forbids processing data you don't have rights to use
- Not using the results of the processing
- Just opening a file and reading it can put you in violation, technically
- We desperately need enforcement mechanisms all the way down the stack
-

Data lifetime management

- TTL on data needs to be baked in at the storage layer
- We're really good at making sure things DON'T get deleted
 - We have erasure codes!
 - We have replication!
 - We have integrity checks!
 - A lot of systems privilege integrity
- We need to also privilege deletion in storage that backs ML data
- We need granular deletion (the "Right to be Forgotten") and it needs to be efficient
- Make sure it actually gets deleted in an information theoretically secure way
- "By default, storage systems interpret deletions as a failure they need to fix."

Privacy from the ground up

- Very granular access controls and policies starting from the lowest levels
 - ACLs by column/region of file, not at a file/table level
 - I want this column to be sensitive (maybe only for this rowset!)
- Auditing
 - Who accessed this? When? From where? What did they see?
- Provenance
 - Where did this data come from? Do I have the legal right to process it? Do I need to delete everything from this source?
 - Where are all the machine learning models which trained on this exact sample?
- Some blue sky ideas
 - Differential privacy primitives at the storage level

Policies for everything

- Easy:
 - Delete all data objects owned by this user
 - Open this file and return its contents
- Harder:
 - Delete all references to this user in every table of a database
 - Open this file and return all of its contents if it doesn't violate a policy
- Really hard:
 - Delete all posts and photos which contain this user
 - Open this file and only return the parts I am allowed to read (here's a function which will tell you if I'm allowed to read them.)

Takeaways

Most ML is not big data

Deep learning is important

But so are graph based algorithms

ML is driving storage constraints for security and privacy more than ever

Thank you!

Questions?

aleatha@humu.com

@aleatha on Twitter