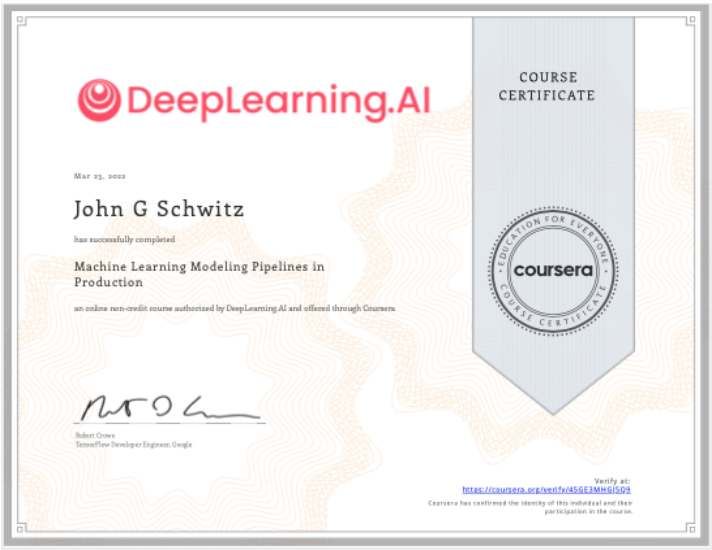
ML Pipeline
May 24, 2022



DeepLearning.AI

COURSE CERTIFICATE

Mar 23, 2022

John G Schwitz

has successfully completed

Machine Learning Modeling Pipelines in Production

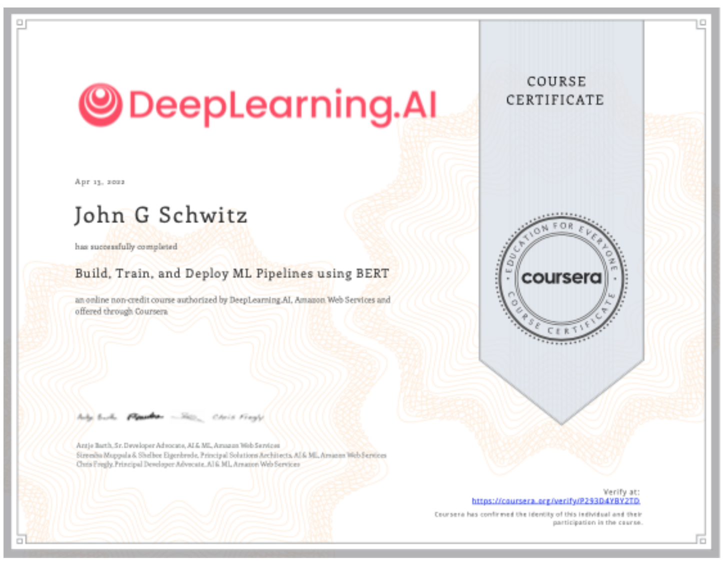
an online non-credit course authorized by DeepLearning.AI and offered through Coursera

Robert Crowe
TensorFlow Developer Engineer, Google

Verify at:
https://coursera.org/verify/45GE3MHGJ5Q9

Coursera has confirmed the identity of this individual and their participation in the course.

coursera
EDUCATION FOR EVERYONE
COURSE CERTIFICATE



DeepLearning.AI

COURSE CERTIFICATE

Apr 13, 2022

John G Schwitz

has successfully completed

Build, Train, and Deploy ML Pipelines using BERT

an online non-credit course authorized by DeepLearning.AI, Amazon Web Services and offered through Coursera

Antje Barth, Sr. Developer Advocate, AI & ML, Amazon Web Services
Sireesha Muppala & Shelbee Eigenbrode, Principal Solutions Architects, AI & ML, Amazon Web Services
Chris Fregly, Principal Developer Advocate, AI & ML, Amazon Web Services

Verify at:
https://coursera.org/verify/P293D4YBY2TD

Coursera has confirmed the identity of this individual and their participation in the course.

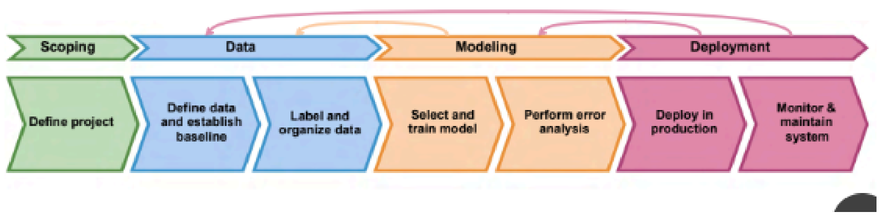
coursera
EDUCATION FOR EVERYONE
COURSE CERTIFICATE

A ML model is only 10% of the code required for Production

The requirements surrounding ML infrastructure

[D. Sculley et. al. NIPS 2015: Hidden Technical Debt in Machine Learning Systems]

The ML project lifecycle



ML PIPELINE Spans: Scoping, Data, Modeling, and Deployment

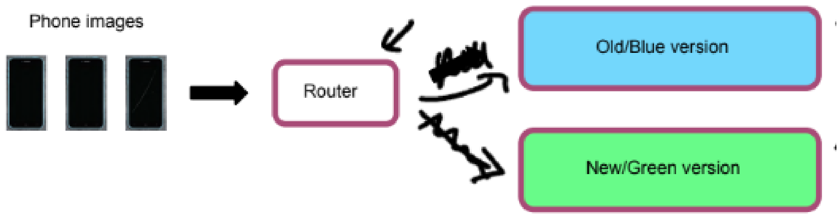
Address: Concept Drift {is user interested in different Financial Information?}, Data Drift {has the training data aged?}

Software engineering issues

Checklist of questions

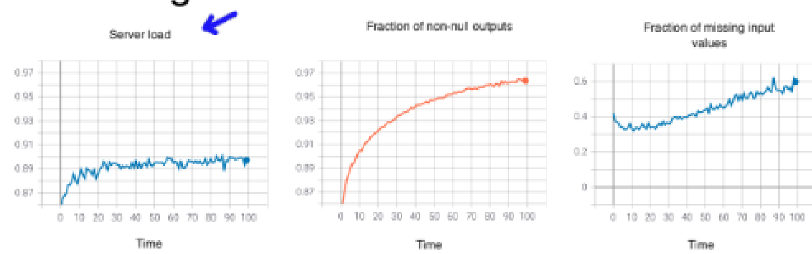
- Realtime or Batch
- Cloud vs. Edge/Browser
- Compute resources (CPU/GPU/memory)
- Latency, throughput (QPS) 50k
- Logging
- Security and privacy

Blue green deployment



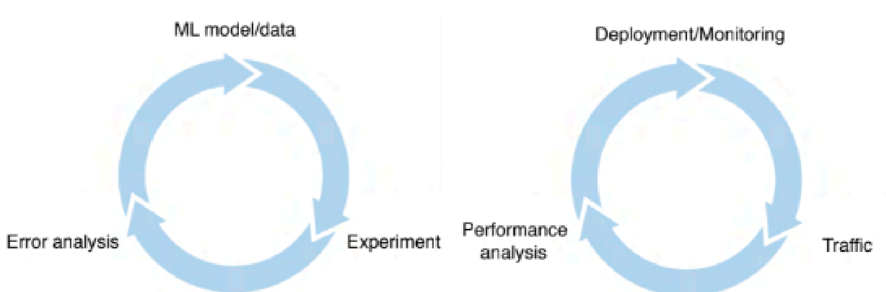
Easy way to enable rollback

Monitoring dashboard



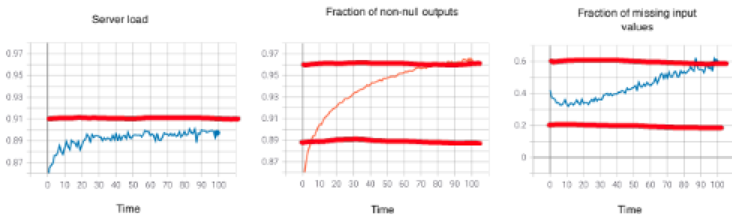
- Brainstorm the things that could go wrong.
- Brainstorm a few statistics/metrics that will detect the problem.
- It is ok to use many metrics initially and gradually remove the ones you find not useful.

Just as ML modeling is iterative, so is deployment



Iterative process to choose the right set of metrics to monitor.

Monitoring dashboard



- Set thresholds for alarms
- Adapt metrics and thresholds over time

Metrics to monitor

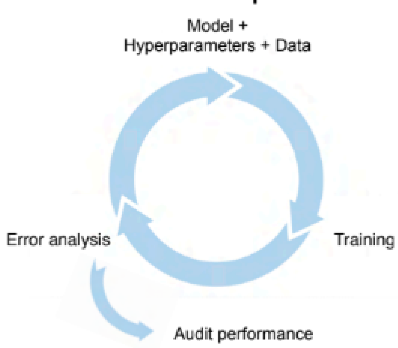
Monitor

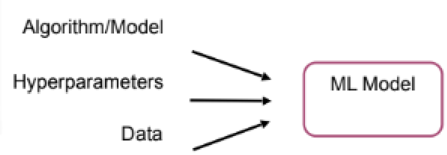
- Software metrics
- Input metrics
- Output metrics

How quickly do they change?

- User data generally has slower drift.
- Enterprise data (B2B applications) can shift fast.

Model development is an iterative process





Performance on key slices of the dataset

Example: ML for loan approval

Make sure not to discriminate by ethnicity, gender, location, language or other protected attributes.

Example: Product recommendations from retailers

Be careful to treat fairly all major user, retailer, and product categories.

Rare classes

Skewed data distribution

99% negative 1% positive

print("0")

Accuracy in rare classes

	Condition	Performance
10,000	Effusion	0.901
	Edema	0.924
	Mass	0.909
~100	Hernia	0.851

Ways to establish a baseline

- Human level performance (HLP)
- Literature search for state-of-the-art/open source
- Older system

Baseline gives an estimate of the irreducible error / Bayes error and indicates what might be possible.

Getting started on modeling

- Literature search to see what's possible.
- Find open-source implementations if available.
- A reasonable algorithm with good data will often outperform a great algorithm with not so good data.

Prioritizing what to work on

Decide on most important categories to work on based on:

- How much room for improvement there is.
- How frequently that category appears.
- How easy is to improve accuracy in that category.
- How important it is to improve in that category.

From Big Data to Good Data

Try to ensure consistently high-quality data in all phases of the ML project lifecycle.

Good data is:

- Cover of important cases (good coverage of inputs x)
- Defined consistently (definition of labels y is unambiguous)
- Has timely feedback from production data (distribution covers data drift and concept drift)
- Sized appropriately

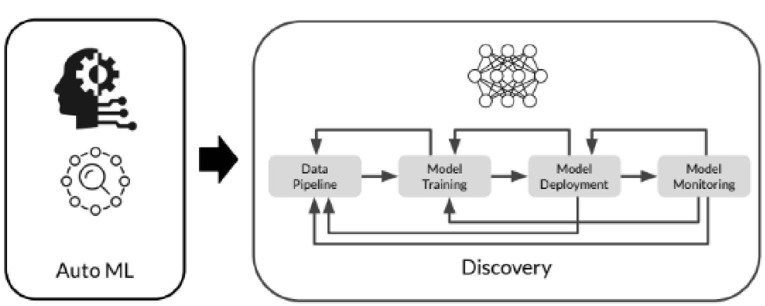
Types of parameters in ML Models

- Trainable parameters:
  - Learned by the algorithm during training
  - e.g. weights of a neural network
- Hyperparameters:
  - set before launching the learning process
  - not updated in each training step
  - e.g: learning rate or the number of units in a dense layer

Manual hyperparameter tuning is not scalable

- Hyperparameters can be numerous even for small models
- e.g shallow DNN:
  - Architecture choices
  - activation functions
  - Weight initialization strategy
  - Optimization hyperparameters such as learning rate, stop condition
- Tuning them manually can be a real brain teaser
- Tuning helps with model performance

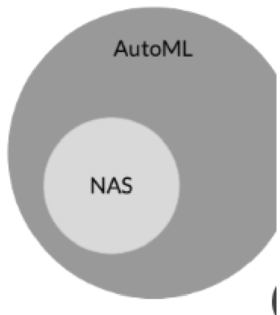
Automated Machine Learning (AutoML)



AutoML automates the entire ML workflow

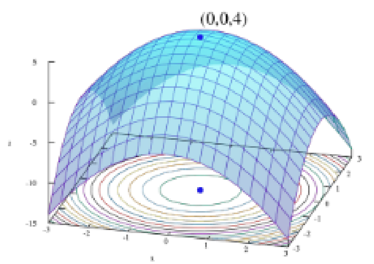
Neural Architecture Search

- **AutoML** automates the development of ML models
- **AutoML** is not specific to a particular type of model.
- Neural Architecture Search (**NAS**) is a subfield of AutoML
- NAS is a technique for automating the design of artificial neural networks (ANN).



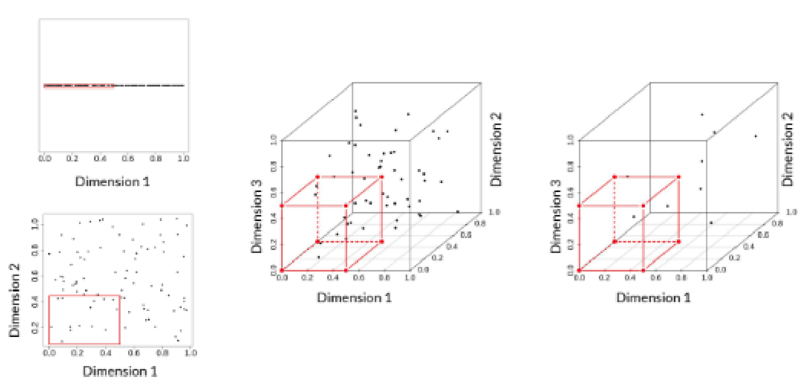
A Few Search Strategies

1. Grid Search
2. Random Search
3. Bayesian Optimization
4. Evolutionary Algorithms
5. Reinforcement Learning

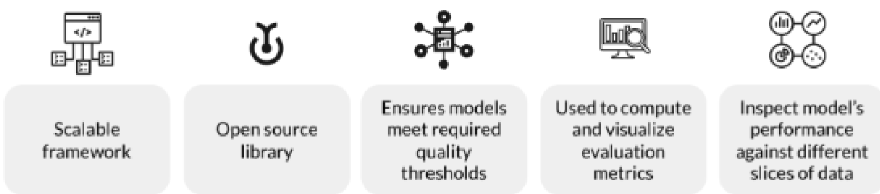


CURSE OF DIMENSIONALITY

Increasing sparsity with higher dimensions



TensorFlow Model Analysis (TFMA)

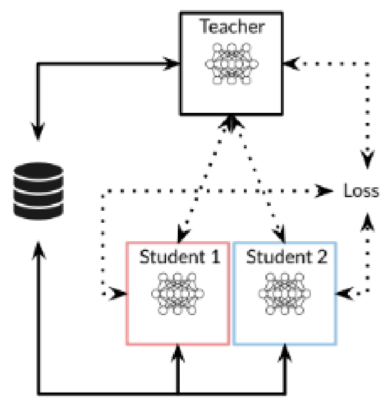


Need for Explainability in AI

1. Models with high sensitivity, including natural language networks, can generate wildly wrong results
2. Attacks
3. Fairness
4. Reputation and Branding
5. Legal and regulatory concerns
6. Customers and other stakeholders may question or challenge model decisions

Knowledge distillation

- Duplicate the performance of a complex model in a simpler model
- Idea: Create a simple *'student'* model that learns from a complex *'teacher'* model



Types of distributed training

- **Data parallelism**: In data parallelism, models are replicated onto different accelerators (GPU/TPU) and data is split between them
- **Model parallelism**: When models are too large to fit on a single device then they can be divided into partitions, assigning different partitions to different accelerators