REALIZING END TO END REPRODUCIBLE MACHINE LEARNING ON KUBERNETES

Suneeta Mall
Senior Data Scientist
Nearmap

nearmap


OUR MISSION
IF WE CHANGE THE WAY PEOPLE VIEW THE WORLD,
WE TRANSFORM THE WAY THEY WORK
Captured: 13 Jun 2018
Sydney, NSW
nearmap

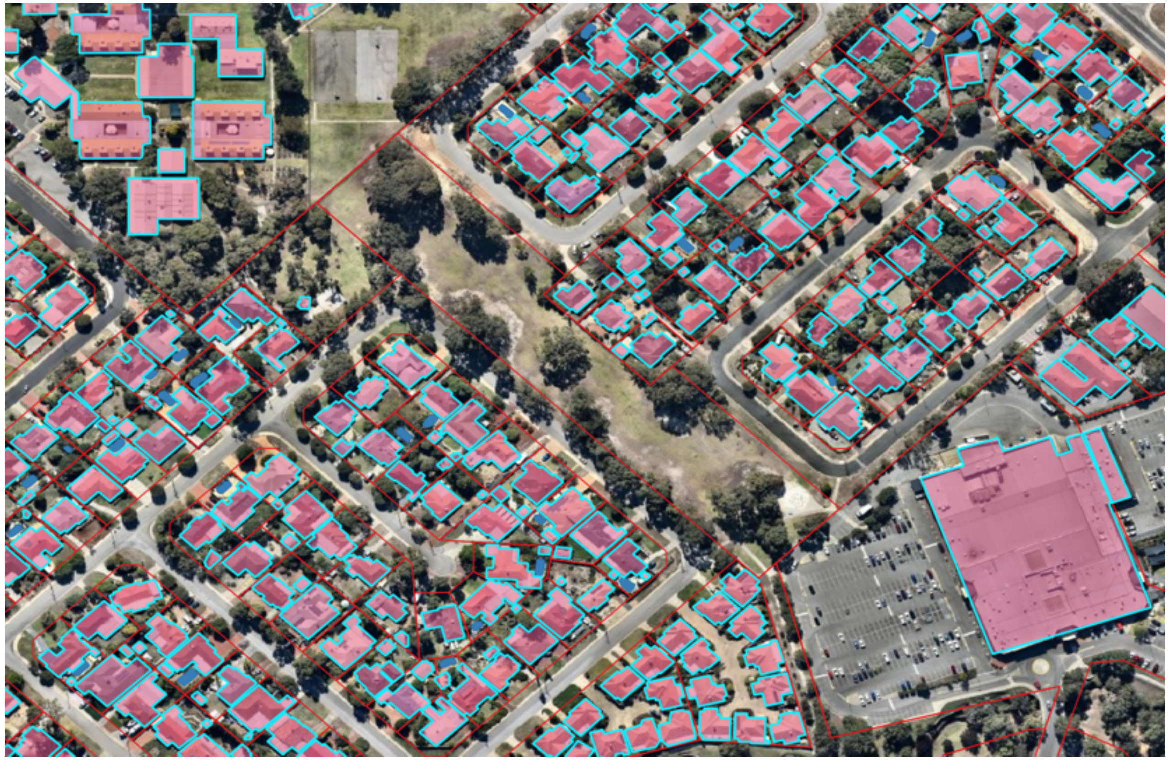
NEARMAP

- Founded in 2006 as a technology and innovation company
- Specialize in high definition aerial imagery delivered via cloud – 2D, 3D, AI and more
- Regularly capture large land areas in US, AU, NZ + CA
- 10,000+ companies leveraging service across the globe
- NEA is a publicly traded stock on the ASX

AI @ NEARMAP

Our team has been doing lots of exciting work building derived content

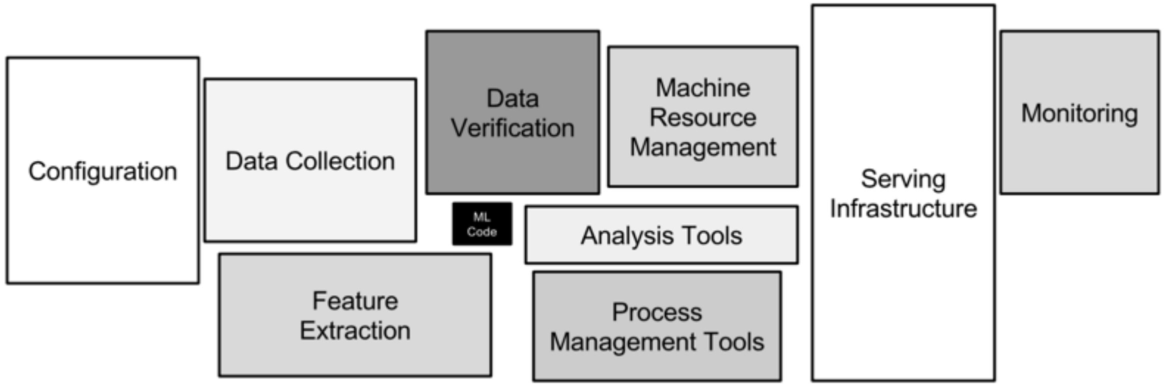
- Building outlines
- Things and stuff detection



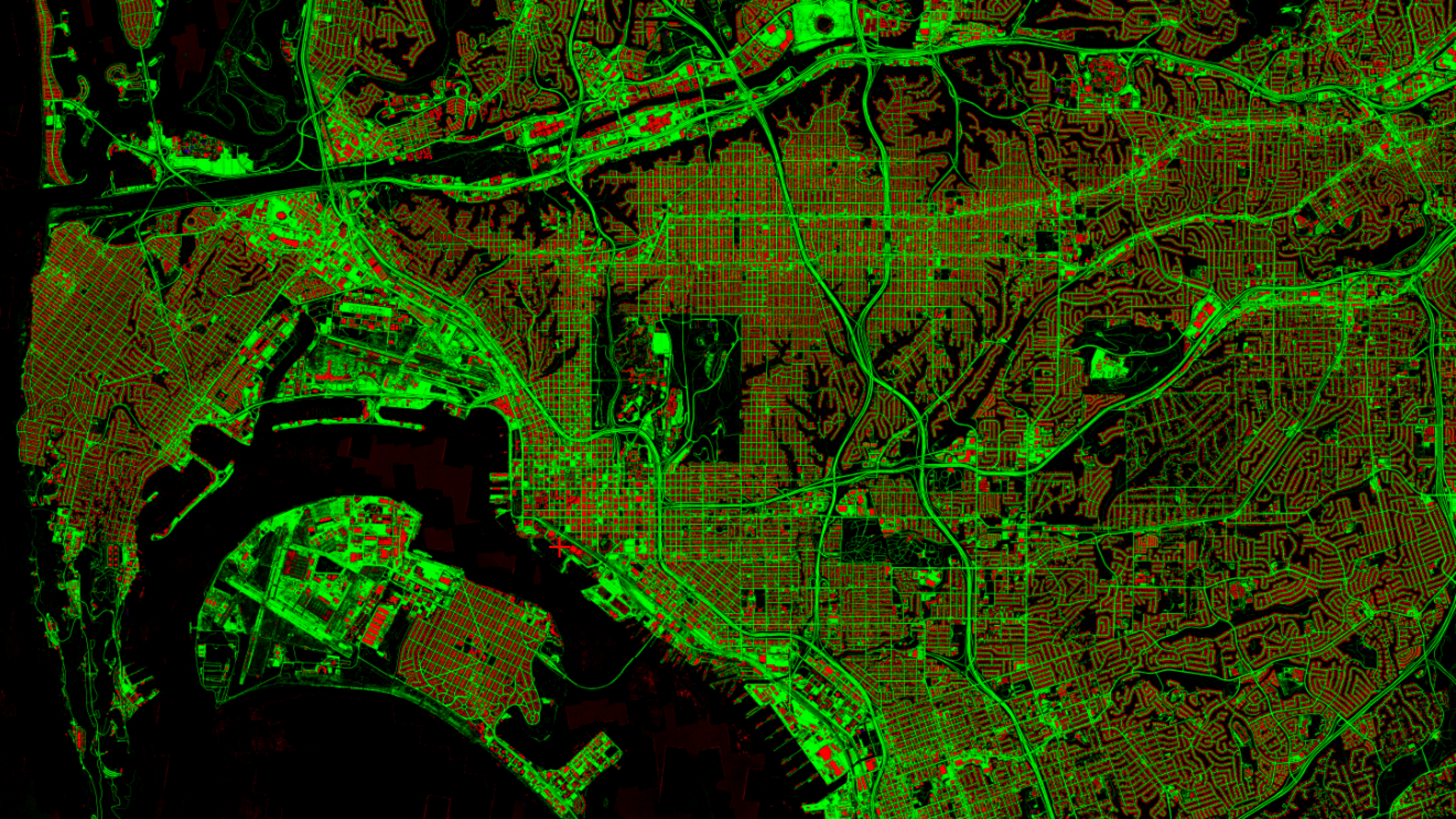
KUBERNETES: AI PLATFORM

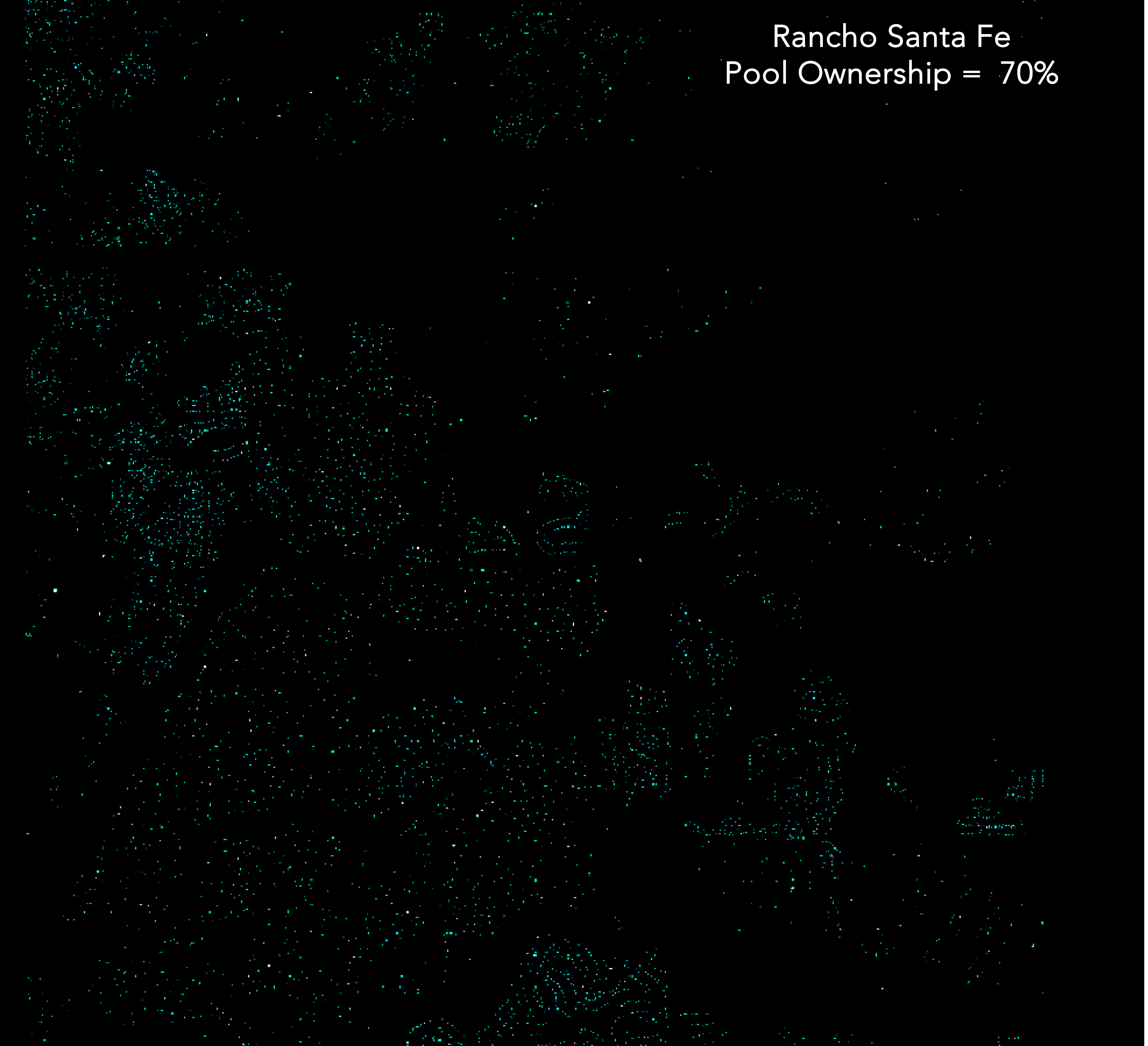
- AI system IS NOT just about model
- Joint effort of Data Scientists, Statisticians, Engineering, Devops & DataOps (MLOps)

- Resiliency, Scale, Platform abstraction and Agnosticism is desired
- Need to simplify orchestration & operations
- Need to provide a declarative ML platform



Sculley *et al.* Hidden Technical Debt in Machine Learning Systems. NIPS (2015)




Rancho Santa Fe
Pool Ownership = 70%


National City
Pool Ownership = 3%

AI @ SCALE

Crunching through petabytes size data, exhausting all K80 spot GPUs across all US data centers of AWS for weeks to produce semantic content on over a million km^2 area at resolution as high as 5cm/pixel in just 2 weeks.

Thursday, December 12 • 10:30 - 10:55

Running Massively Parallel Deep-learning Inference Pipelines on Kubernetes - Suneeta Mall & Martin Abeleda, Nearmap

Kubernetes Forum Sydney

nearmap

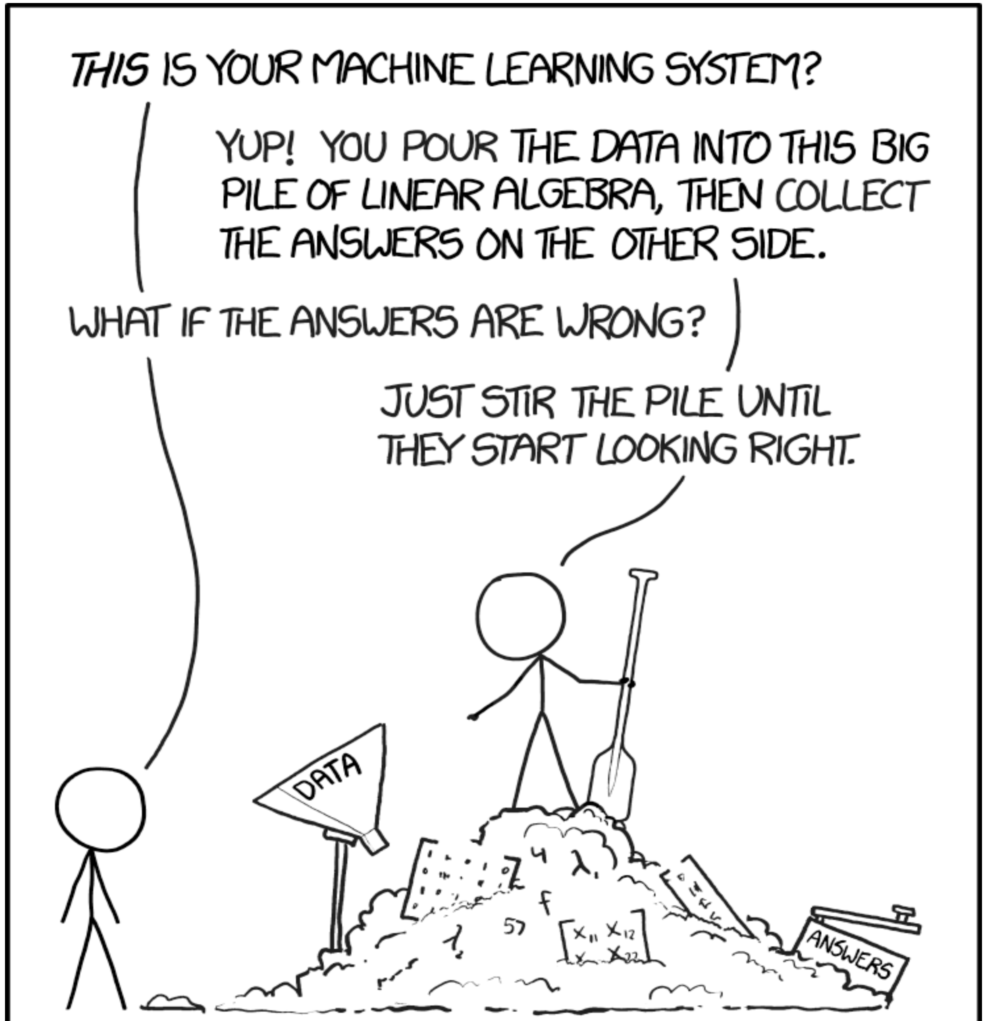
AGENDA

- Machine learning
- Why reproducibility
- Challenges in reproducible AI/ML
- How much reproducibility do we need
- What are some of the tools and techniques
- Realizing reproducible AI/ML seamlessly on K8s
- Robust models: replicability an extension to reproducibility
- Questions

New York, USA

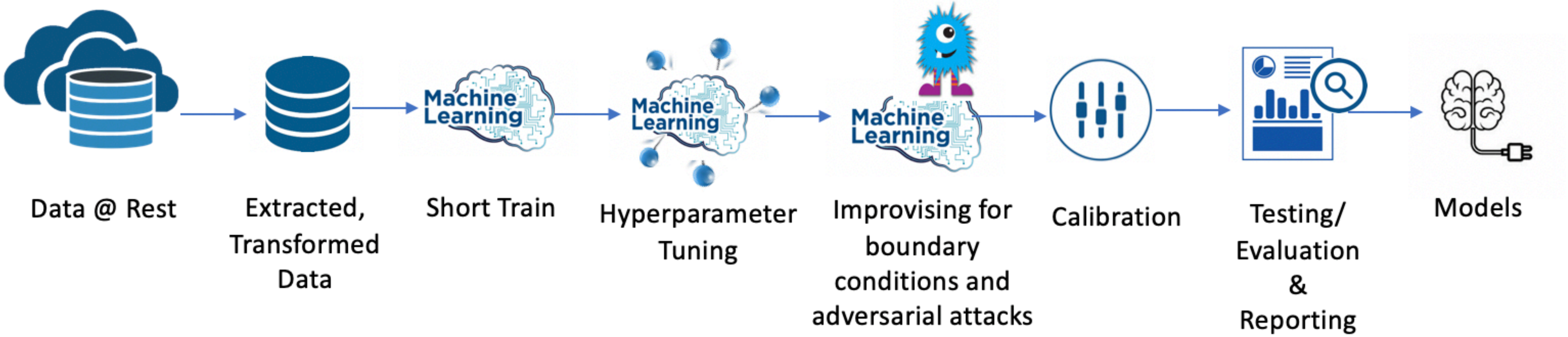
nearmap

MACHINE LEARNING

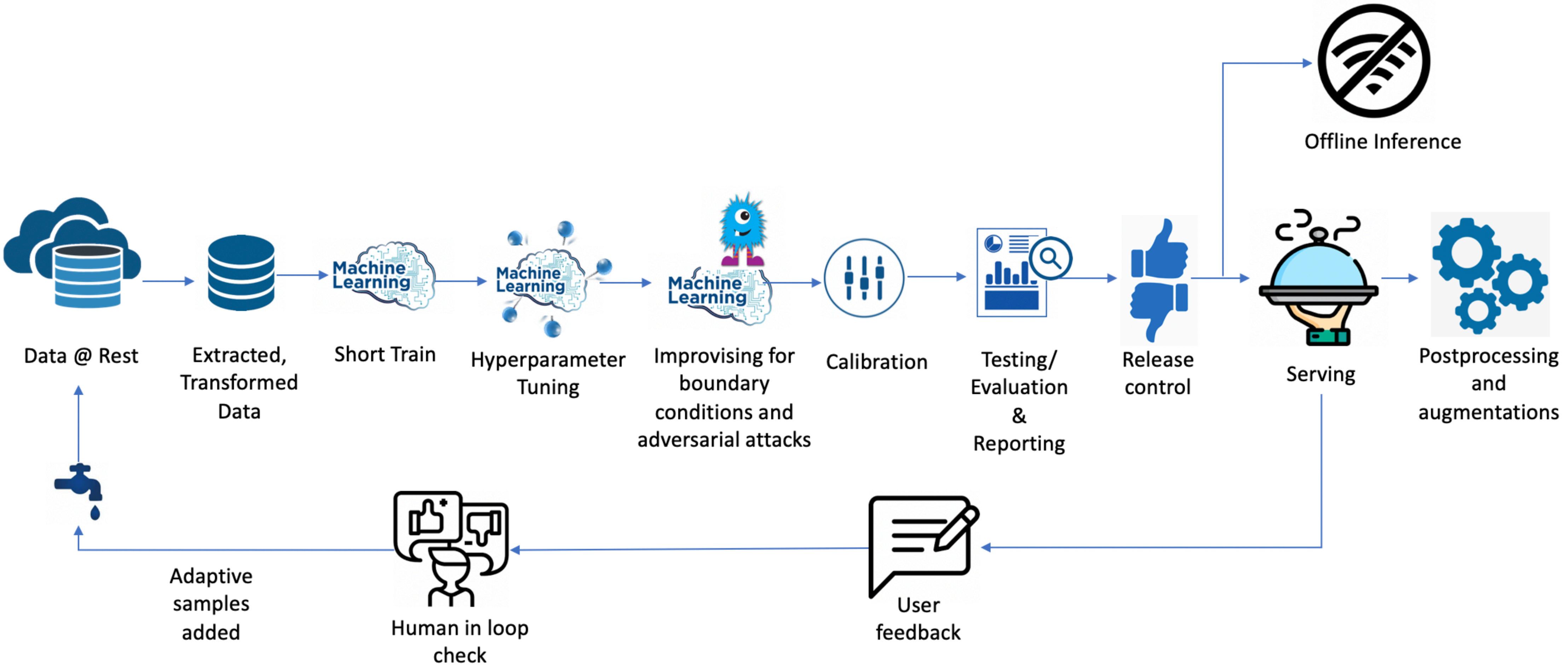


https://xkcd.com/1838/

STANDARD ML WORKFLOW



REALITY



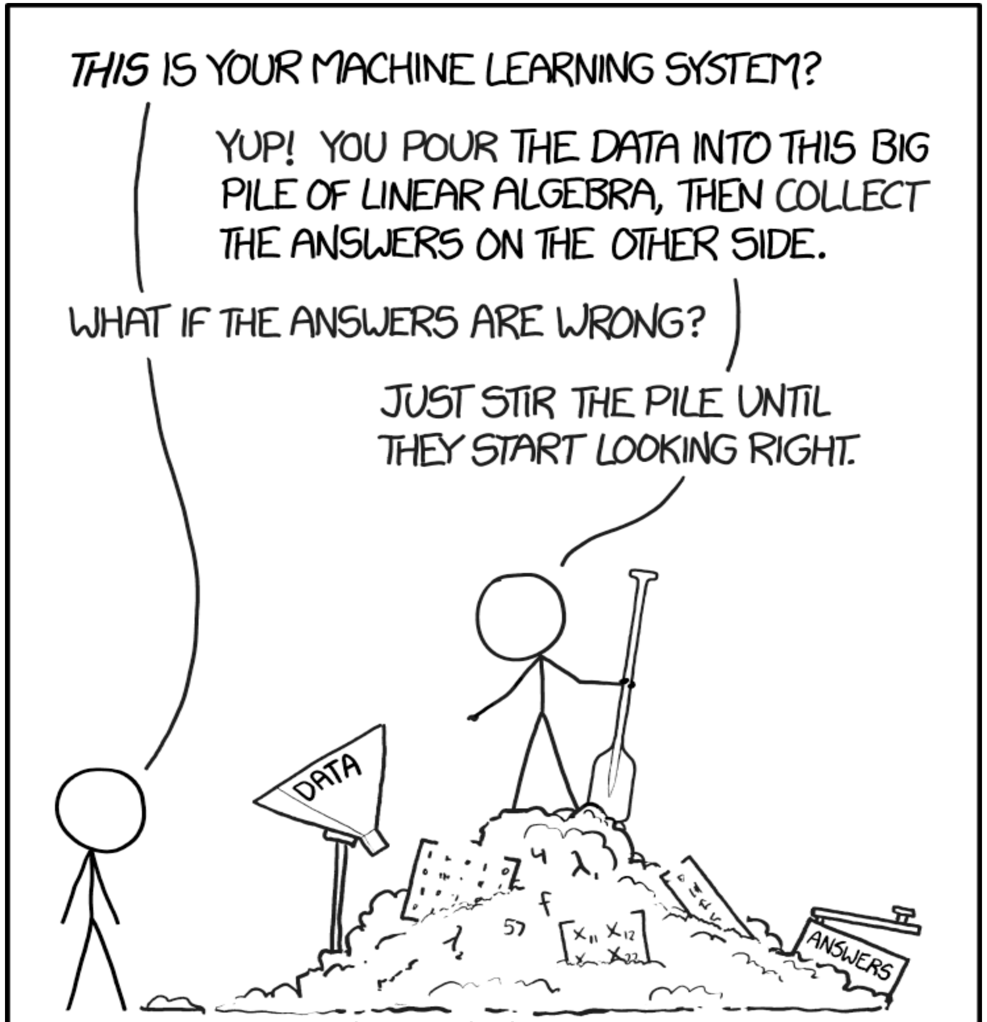
nearmap

WHY REPRODUCIBILITY?

Sun City, Arizona, USA

nearmap

WHY REPRODUCIBILITY: TO UNDERSTAND

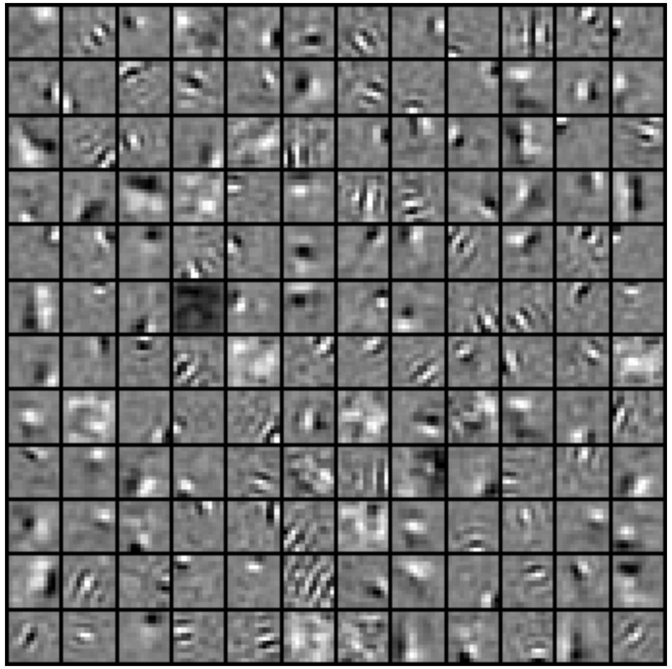


https://xkcd.com/1838/

To debug, understand and explain the deductions

WHY REPRODUCIBILITY: TO UNDERSTAND

- To <u>debug, understand and explain</u> the deductions
- Hello Deep learning



Erhan *et al.* “Visualizing higher-layer features of a deep network”
2009

10 Years of research

Olah *et al.* “The Building Blocks of Interpretability”
2018

WHY REPRODUCIBILITY: CREDIBILITY

- Users expectations: End user expects answers to <u>verifiable, reliable, unbiased and ethical</u>
- Governance: reproducible, traceable, and verifiable

<u>"Good results are not enough, Making them easily reproducible also makes them credible"</u>

- Lecun @ International Solid State Circuit Conference in San Francisco, 2019

nearmap

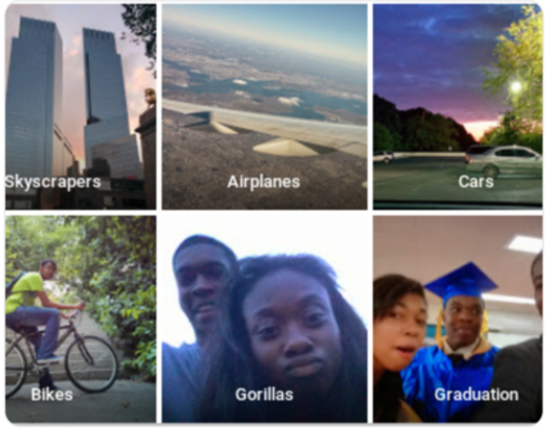
WHY REPRODUCIBILITY: CORRECTNESS

If anything can go wrong, it will Murphy's law

Amazon scraps secret AI recruiting tool that showed bias against women

jackyalcine really couldn't give a f...
@jackyalcine
Follow

Google Photos, y'all fucked up. My friend's not a gorilla.



6:22 pm - 28 Jun 2015

IBM's Watson gave unsafe recommendations for treating cancer

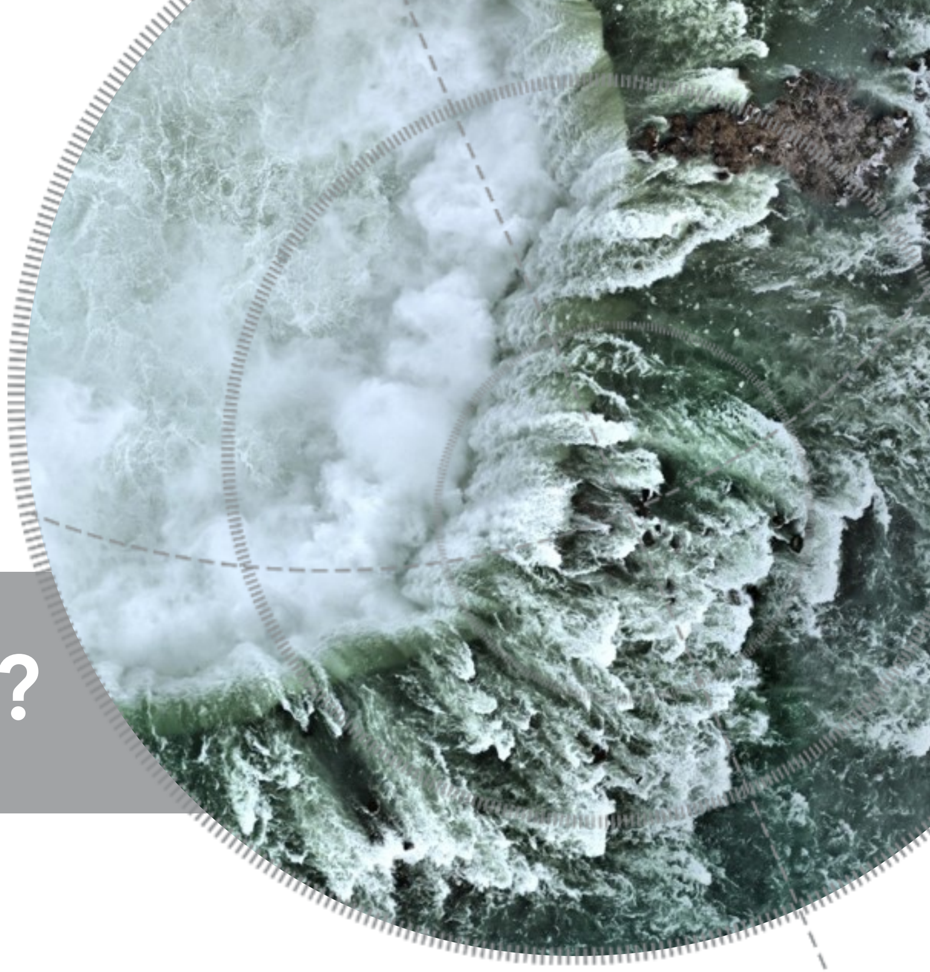
WHY REPRODUCIBILITY: EXTENSIBILITY

The Foundation need be reproducible & reliable!

Roofs
Trees
Driveways
Pools

4942.57 m^2

CHALLENGES IN REPRODUCIBLE ML?



Captured: Niagara Falls, CANADA

CHALLENGES IN REPRODUCIBLE AI/ML

- Hardware
  - Variations from GPU architecture
  - CPU multithreading variations (SESS)
- Software
  - Unintended behaviours due to dependencies
  - Floating Point Precision
- Algorithmic
  - Dropouts
  - Random initializations
  - Shuffles
  - Random Augmentations
- Practices/Process
  - Randomizations
  - Iterative Exploratory
- Data
  - Incomplete Demographics
  - Poisoned data

nearmap

CHALLENGES: HARDWARE

- Different GPU architectures (Stream Multiprocessing)
- Even parallelism on CPU may give different results
  - Intra and Inter ops threads parallelism

"Consistency of Floating Point Results or Why doesn't my application always give the same answer?"

https://www.nccs.nasa.gov/images/FloatingPoint_consistency.pdf
Corden (2008) Intel

CHALLENGES: SOFTWARE

2.7. Reproducibility (determinism)

By design, most of cuDNN's routines from a given version generate the same bit-wise results across runs when executed on GPUs with the same architecture and the same number of SMs. However, bit-wise reproducibility is not guaranteed across versions, as the implementation of a given routine may change. With the current release, the following routines do not guarantee reproducibility because they use atomic operations:

- `cudnnConvolutionBackwardFilter` when `CUDNN_CONVOLUTION_BWD_FILTER_ALGO_0` or `CUDNN_CONVOLUTION_BWD_FILTER_ALGO_3` is used
- `cudnnConvolutionBackwardData` when `CUDNN_CONVOLUTION_BWD_DATA_ALGO_0` is used
- `cudnnPoolingBackward` when `CUDNN_POOLING_MAX` is used
- `cudnnSpatialTfSamplerBackward`

https://docs.nvidia.com/deeplearning/sdk/cudnn-developer-guide/index.html#reproducibility

"Determinism in deep learning" By Duncan Riach @ GTC 2019

https://drive.google.com/file/d/18pmjeiXWqzHWB8mM2mb3kjN4JSOZBV4A/view

CHALLENGES: SOFTWARE

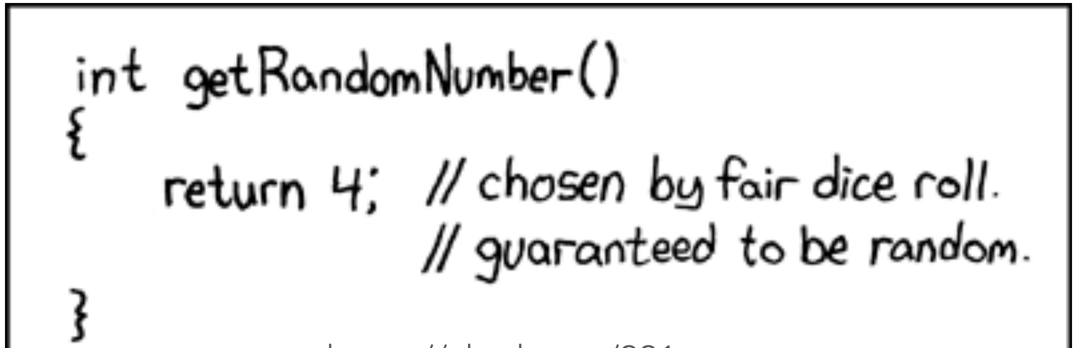
Story of *Pyproj* (a geospatial transform library) upgrade from V1.9.6 to V2.4.0

Location calculation for San Diego Convention Centre = Somewhere in Miramar off golf course

True story: https://github.com/pyproj4/pyproj/issues/470

CHALLENGES: ALL THE THINGS RANDOMNESS

- Algorithmic
- Dropouts
- Random initializations
- Random augmentations
- Random noise introduction (adversarial robustness)
- Shuffles



https://xkcd.com/221

Get a grip with the (random) seed!

```
os.environ['PYTHONHASHSEED'] = str(seed)
random.seed(seed)
tf.random.set_seed(seed)
np.random.seed(seed)
tf.keras.layers.Dropout(x, seed=SEED)
```

nearmap

CHALLENGES: DATA

- **Change Anything Changes Everything** principle
- No inputs are ever really independent

(Scully, 2015)

- How to get back to same data in exact same sequence to diagnose & resolve:
  - Data poisoning
  - Under/Over-represented data (inappropriate demographic)

gerry
@geraldmellor
Follow

"Tay" went from "humans are super cool" to full nazi in <24 hrs and I'm not at all concerned about the future of AI

https://twitter.com/geraldmellor/status/712880710328139776

THEN WE HAVE …

- A model is rarely deployed twice (Talby, 2018)*
- Concept drift
- Continual learning – full automation and governance

First prototype car

First car to do road trip

Automobiles

Self driving Solar powered cars

1807 1858 1909 1980 2018

*https://www.oreilly.com/radar/lessons-learned-turning-machine-learning-models-into-real-products-and-services/

HOW MUCH REPRODUCIBILITY DO YOU NEED?

As long as it can be explained, understood, reclaimed!

"…offers a road map to reach the same conclusions "

*Dodge**

or

*https://www.wired.com/story/artificial-intelligence-confronts-reproducibility-crisis/

HOW TO REALIZE REPRODUCIBILITY

Because CACE is real

1) Reproducible ML code

2) ~~Don't change anything~~

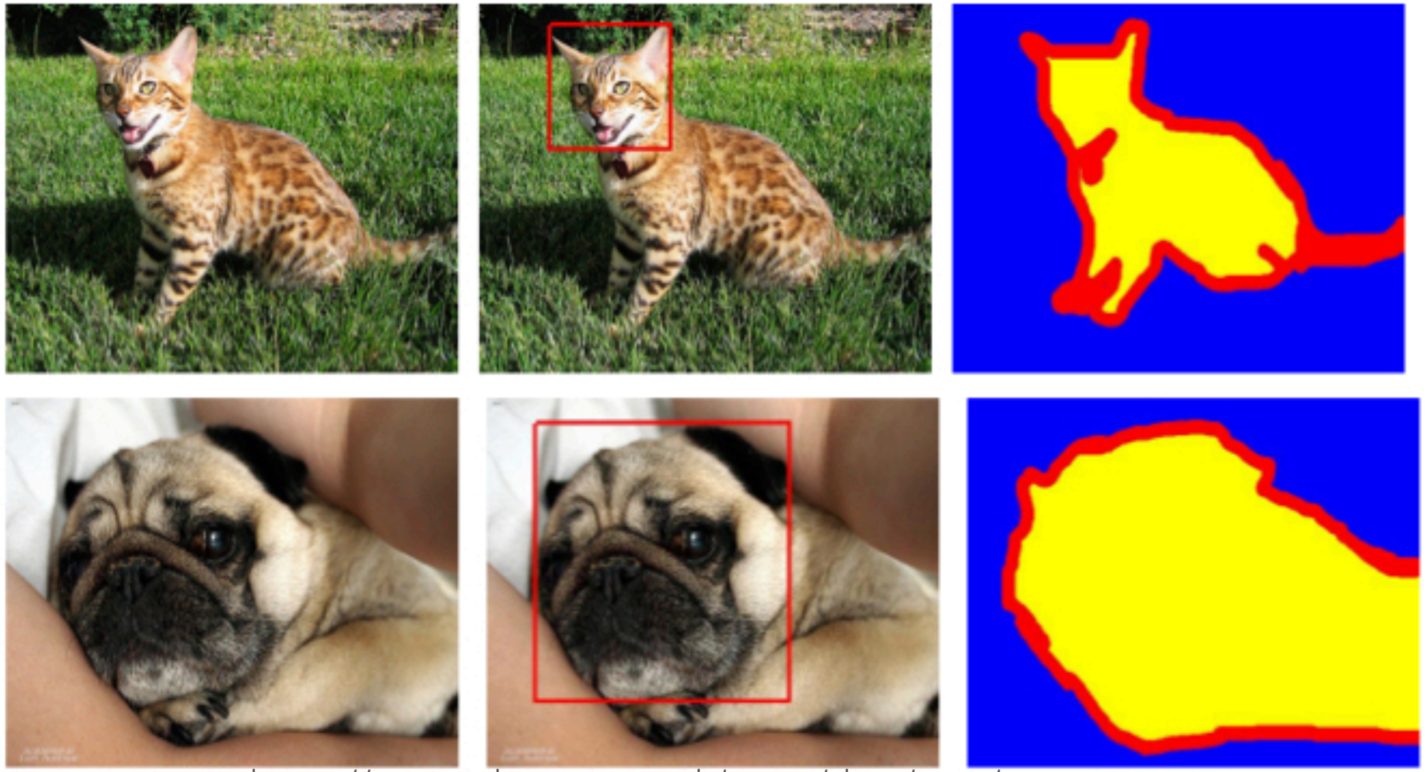
2) Version Control everything!

https://xkcd.com/1597/

REFERENCE EXAMPLE APP: END TO END ML ON KUBERNETES

Sample app: https://github.com/suneeta-mall/e2e-ml-on-k8s.git

Oxford Pet Dataset:



https://www.robots.ox.ac.uk/~vgg/data/pets/

1. REPRODUCIBLE ML CODE

Algorithmic

Software

Captured: Washington, USA

nearmap

GOTCHAS OF CODE

- Code is *version controlled*
- Reproducible runtime – *pinned* libraries
- Smart randomness
- Rounding precision & overflows
- Dependent library's behavior *aware*

"Backward pass of broadcasting on GPU is non-deterministic"
https://github.com/tensorflow/tensorflow/issues/2652

nearmap

ACHIEVING 100% REPRODUCIBILITY

See train.py and stack:

@ https://github.com/suneeta-mall/e2e-ml-on-k8s.git

1. Every randomness is seeded
2. Libraries pinned
3. And I have

```python
def set_seeds(seed=SEED):
    os.environ['PYTHONHASHSEED'] = str(seed)
    random.seed(seed)
    tf.random.set_seed(seed)
    np.random.seed(seed)
```

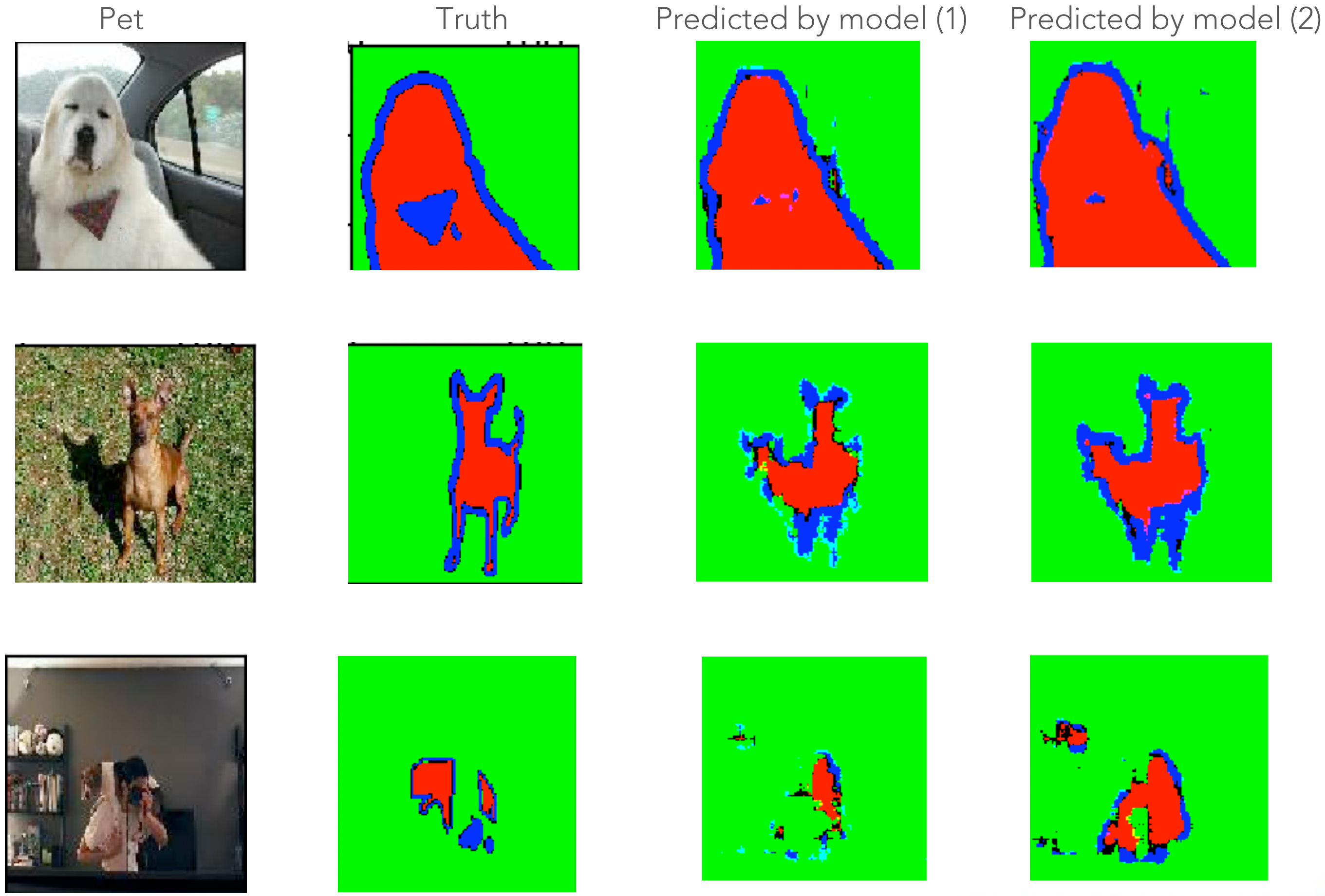
```python
def set_global_determinism(seed=SEED, fast_n_close=False):
    set_seeds(seed=seed)
    if fast_n_close:
        return
    os.environ['TF_DETERMINISTIC_OPS'] = '1'
    os.environ['TF_CUDNN_DETERMINISTIC'] = '1'
    tf.config.threading.set_inter_op_parallelism_threads(1)
    tf.config.threading.set_intra_op_parallelism_threads(1)
    from tfdeterminism import patch
    patch()
```

But saying training is snail-ish is an understatement

28 mins vs 1 hr 45 mins

FORGETTING TO SET A SEED

1. Trained with reproducible code (train.py)*
2. Trained with same code as (1) but unseeded dropout layer



* https://github.com/suneeta-mall/e2e-ml-on-k8s.git

2 VERSIONING CONTROL

Hardware

Software

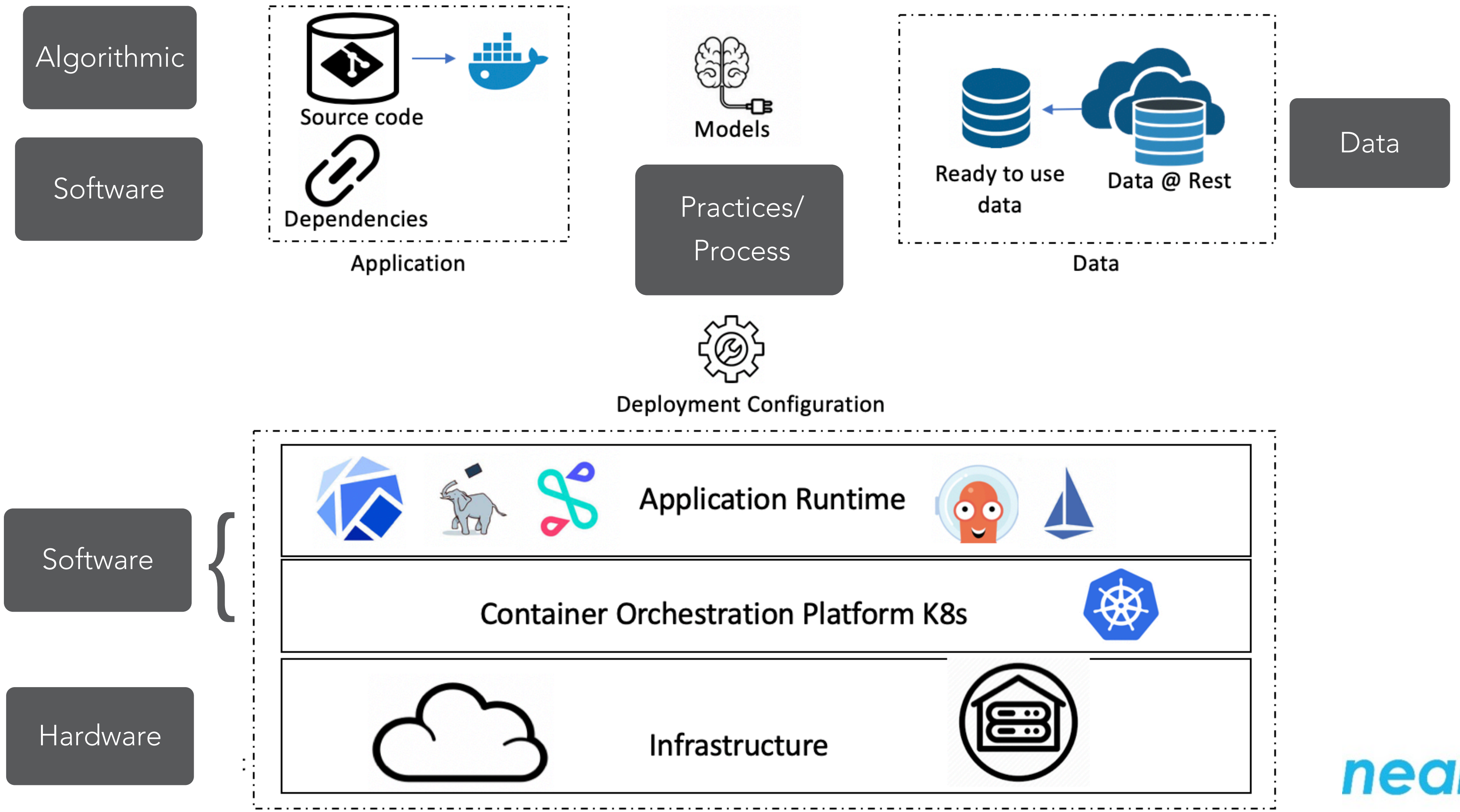
Algorithmic

Process

Data

Captured: Perth, AUSTRALIA

WHAT TO VERSION CONTROL?



2.1 VERSIONING ENVIRONMENT

Hardware

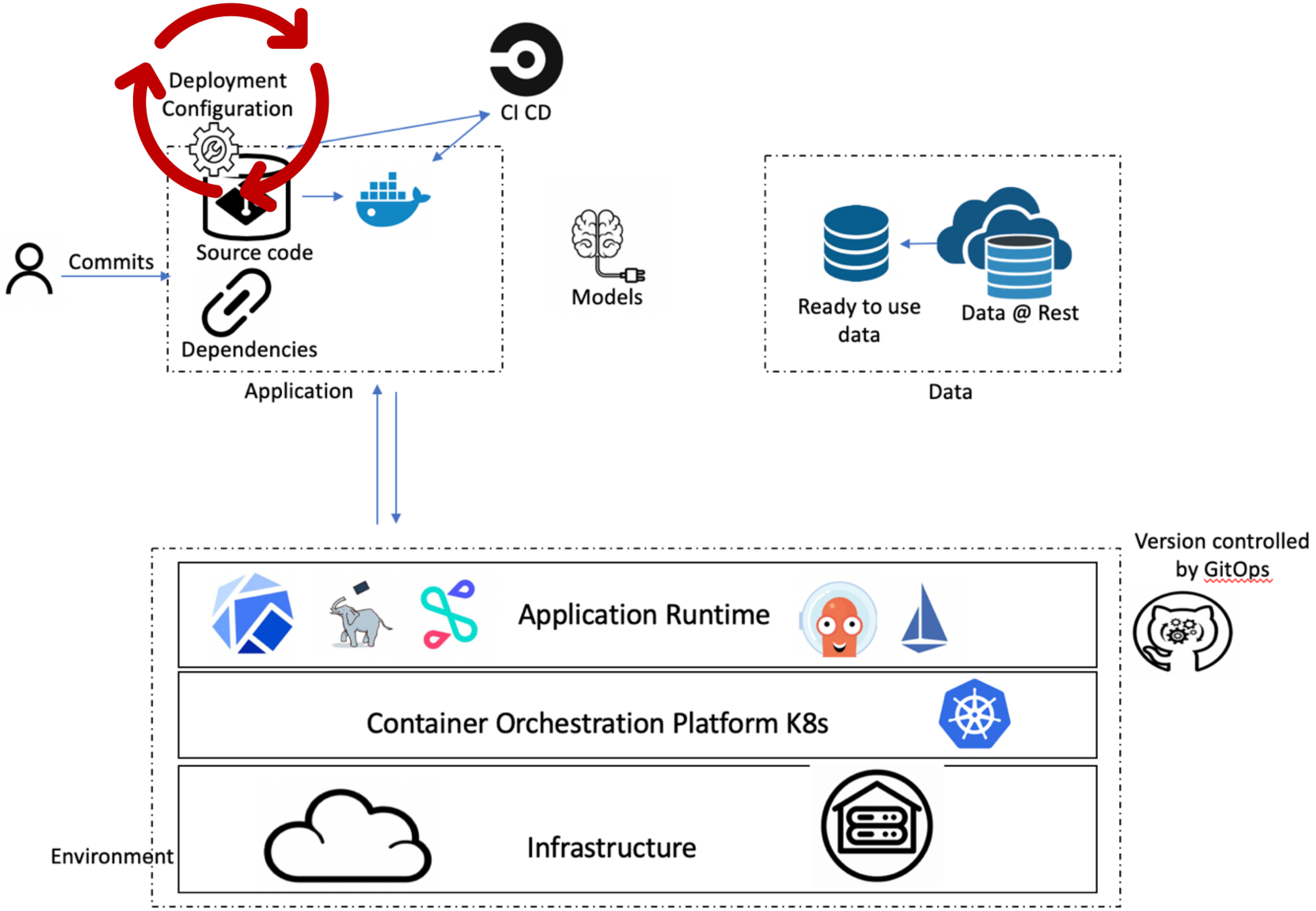
Software

Captured: Perth, AUSTRALIA

nearmap

GITOPS: VERSIONING ENVIRONMENT

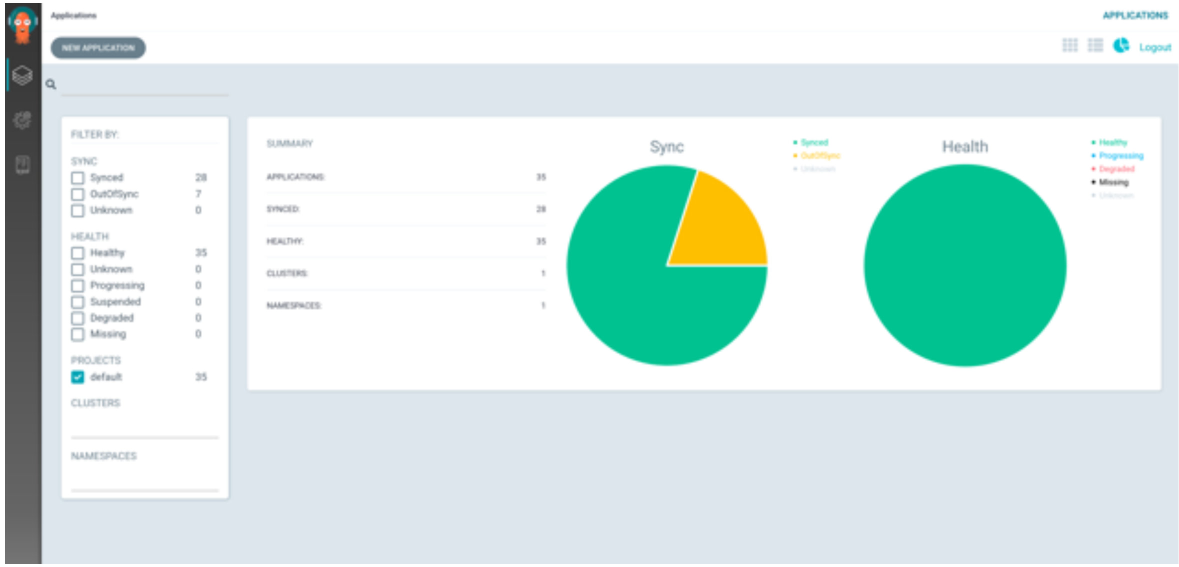
Standard continuous integration and continuous delivery flow with gitops



GITOPS WITH ARGOCD

BYO Kubernetes cluster, install Argo CD & then:

$ kubectl apply –f cluster-conf/e2e-ml-argocd-app.yaml*



Using Kustomize, configures cluster for:

- Jupyter
- Training frameworks TFJob, TorchJob etc.
- DAG pipelines Kubeflow, Pachyderm, Argo
- HP Tuning: Katib, Ray
- Serving (Seldon, TFServe etc.)
- Istio (Service Mesh)

* https://github.com/suneeta-mall/e2e-ml-on-k8s.git

2.2 VERSIONING: WORKFLOW & DATA

Algorithmic

Software

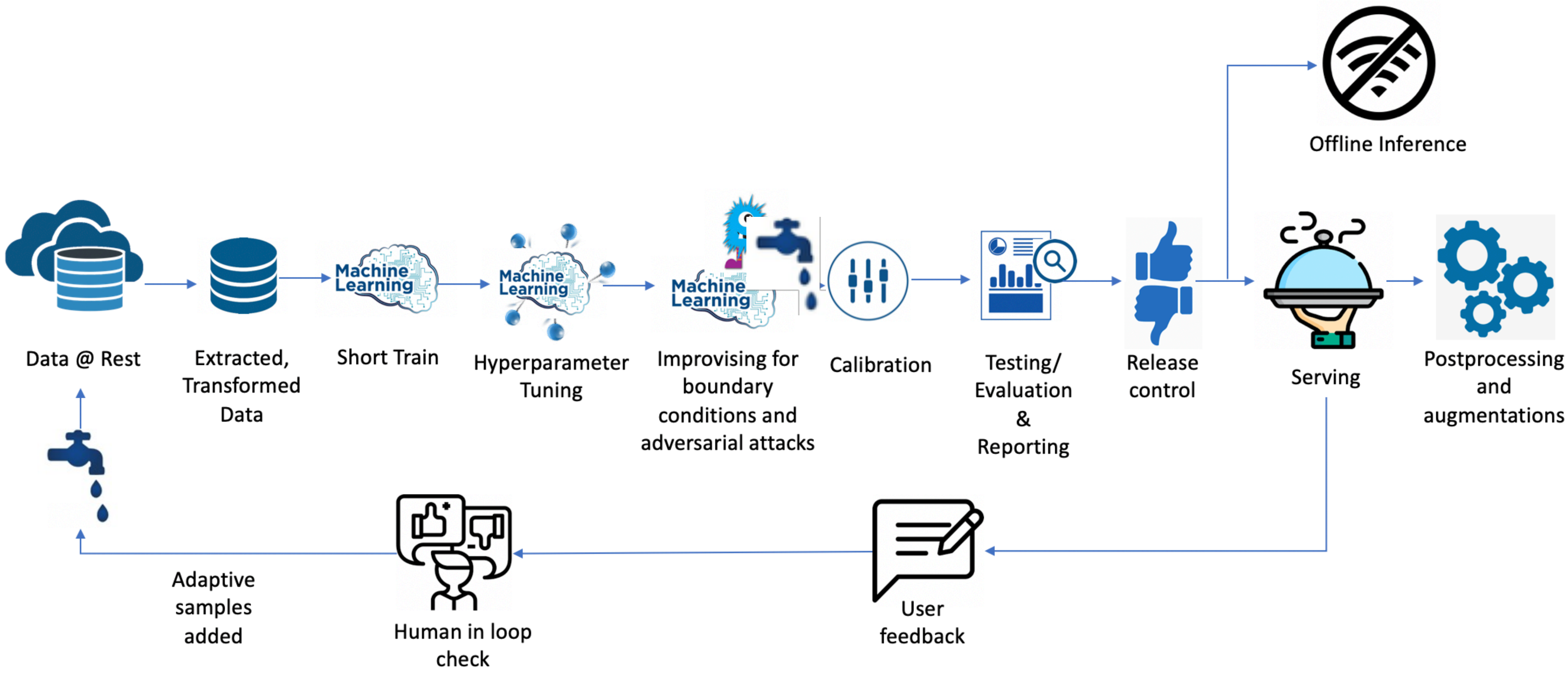
Process

Data

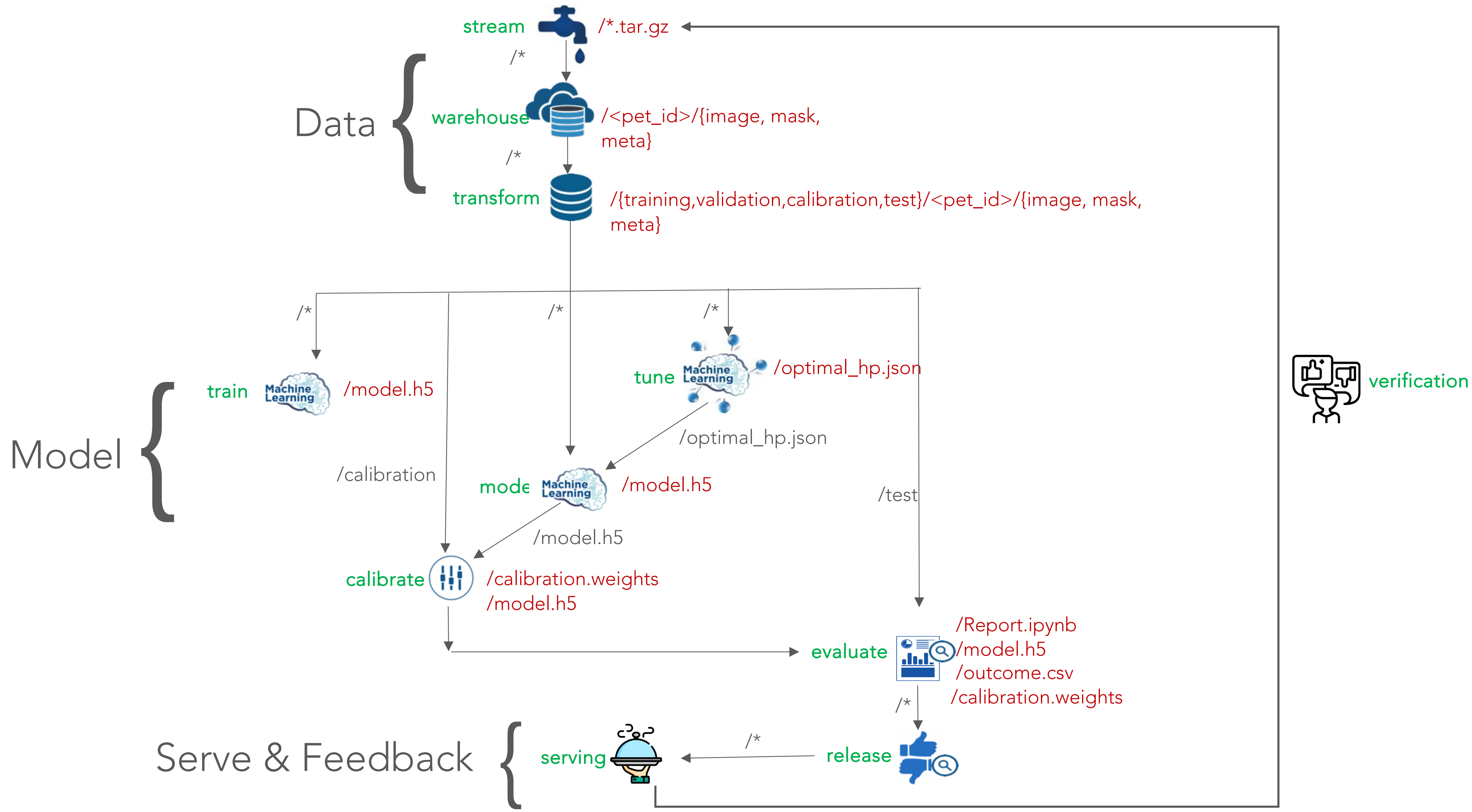
Captured: Dallas, USA

nearmap

REALITY



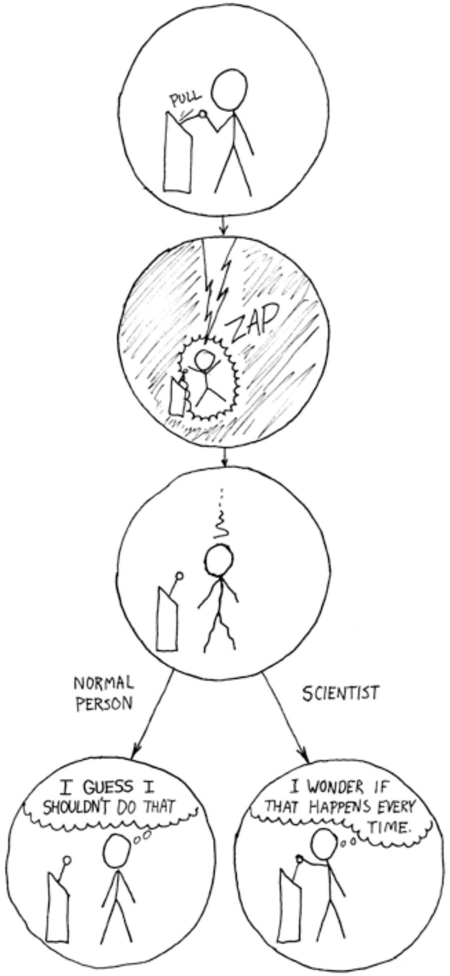
Offline Inference
Machine Learning
Machine Learning
Machine Learning
Data @ Rest
Extracted, Transformed Data
Short Train
Hyperparameter Tuning
Improvising for boundary conditions and adversarial attacks
Calibration
Testing/ Evaluation & Reporting
Release control
Serving
Postprocessing and augmentations
Adaptive samples added
Human in loop check
User feedback


stream
/*.tar.gz
/*
Data
warehouse
/<pet_id>/{image, mask, meta}
/*
transform
/{training,validation,calibration,test}/<pet_id>/{image, mask, meta}
/*
/*
/*
Model
train
/model.h5
tune
/optimal_hp.json
/optimal_hp.json
/calibration
mode
/model.h5
/test
/model.h5
calibrate
/calibration.weights
/model.h5
evaluate
/Report.ipynb
/model.h5
/outcome.csv
/calibration.weights
/*
verification
Serve & Feedback
serving
/*
release

PACHYDERM: FOR PROVENANCE

- Git like data repository
- Automated repositories that `act`
- Pipeline DAG like processes with provenance across graph input, transformation spec, output
- Runs on K8s as backbone

Ref https://github.com/pachyderm/pachyderm



https://xkcd.com/242

KUBEFLOW: ML TOOLKIT

- Take away the pain of infrastructure
- Deployments of machine learning (ML) workflows on Kubernetes simple, portable and scalable
- Enables declarative ML bringing together open source ML framework.
    - Training
    - Tuning (Katib)
    - Serving
    - Pipelines
    - Notebook (Jupyter)

DON'T GEL WELL - YET

- Kubeflow is native K8s but Pachyderm is not!
- Lack of operator supports makes their integration harder

vs

TUNING WITH PACHYDERM

In container tuning:

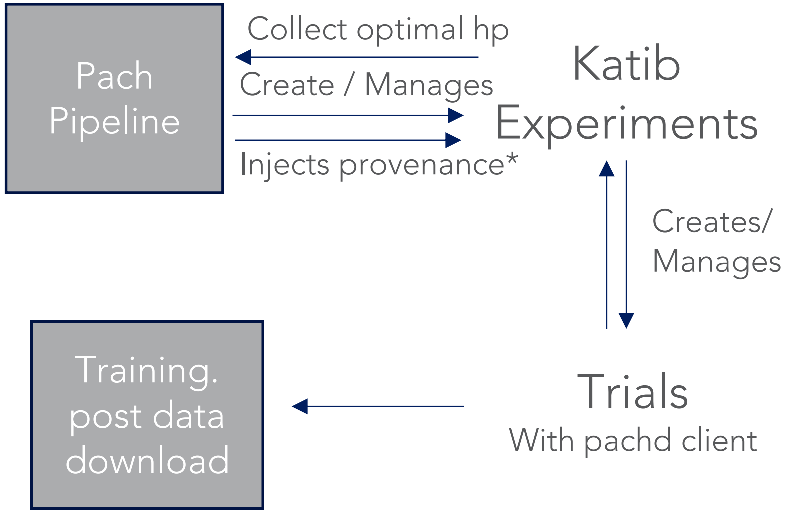
```yaml
`input:
  pfs:
    glob: "/"
    repo: transform
pipeline:
  name: tune
transform:
  cmd:
  - "/bin/bash"
  image: suneetamall/e2e-ml-on-k8s:6
  stdin:
  - "python tune.py --input /pfs/transform --output /pfs/out"
resource_requests:
  memory: 4G
  cpu: 1
datum_tries: 2
```

*cluster-conf/k8s/pipelines/pachyderm-specs.yaml

* https://github.com/suneeta-mall/e2e-ml-on-k8s.git

PACHYDERM IN CONJUNCTION WITH KUBEFLOW

HP Tune with Katib




```yaml
input:
  pfs:
    glob: "/"
    empty_files: true
    repo: transform
pipeline:
  name: tune-kf
transform:
  cmd:
  - "/bin/bash"
  image: suneetamall/e2e-ml-on-k8s:1
  stdin:
  - "python tune_katib.py --input /pfs/transform --output /pfs/out"
pod_spec: '{"serviceAccount": "ml-user", "serviceAccountName": "ml-user"}'
datum_tries: 2
```

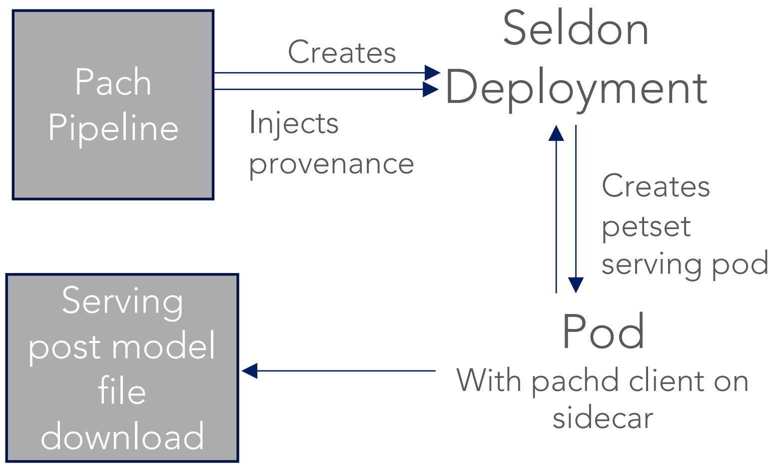
- PACH_JOB_ID
- PACH_OUTPUT_COMMIT_ID
- <input>_COMMIT

e2e-ml-on-k8s@extend_pachyderm-specs-with-kubeflow.yaml

* https://docs.pachyderm.com/reference/pipeline_spec/#environment-variables

RELEASE & SERVING WITH SELDON

- Push model for serving based on evaluation report (release.py)

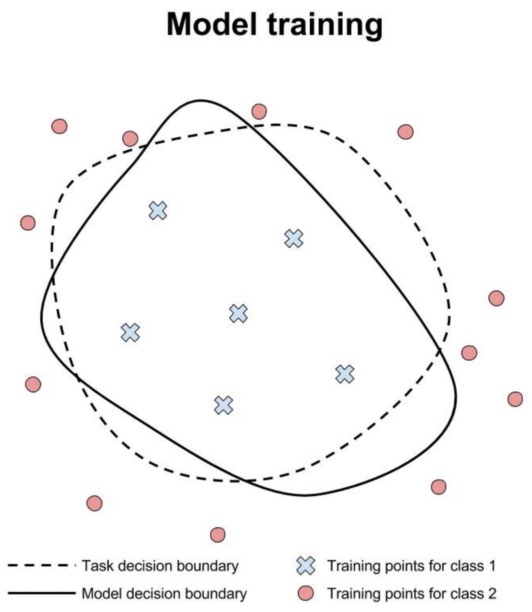


REPLICABILITY: MODEL ROBUSTNESS

- Robust model
  - Cleverhans
  - NSL
  - Foolbox and more ..
- Models generalization across architectures and training sets

  (Explaining and harnessing adversarial examples - Szegedy et al.)
- *Not make confident mistakes*

  (Unrestricted Adversarial Examples - Goodfellow et al.)



http://www.cleverhans.io

SOME TOOLS

DVC + mlflow

https://dvc.org/

https://mlflow.org/

https://martinfowler.com/articles/cd4ml.html

THANK YOU

https://www.nearmap.com/au/en/aerial-view-maps-about-us/geospatial-data-careers

https://www.youtube.com/watch?v=NWhYkjIRvVI