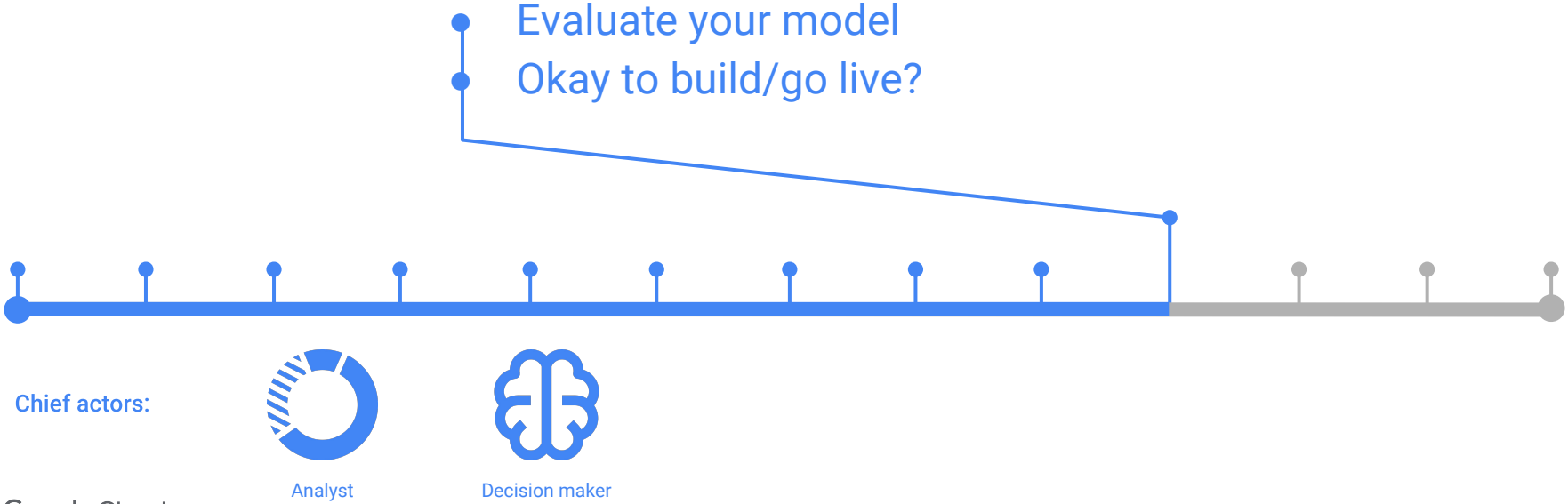
Deep Learning Walkthrough - 09

Code in github.com/google-aai/sc17

Cassie Kozyrkov
Chief Decision Scientist, Google Cloud
GitHub: kozyrkov; Twitter: @quaesita

Google Cloud

Step 9 | Test your model



Google Cloud

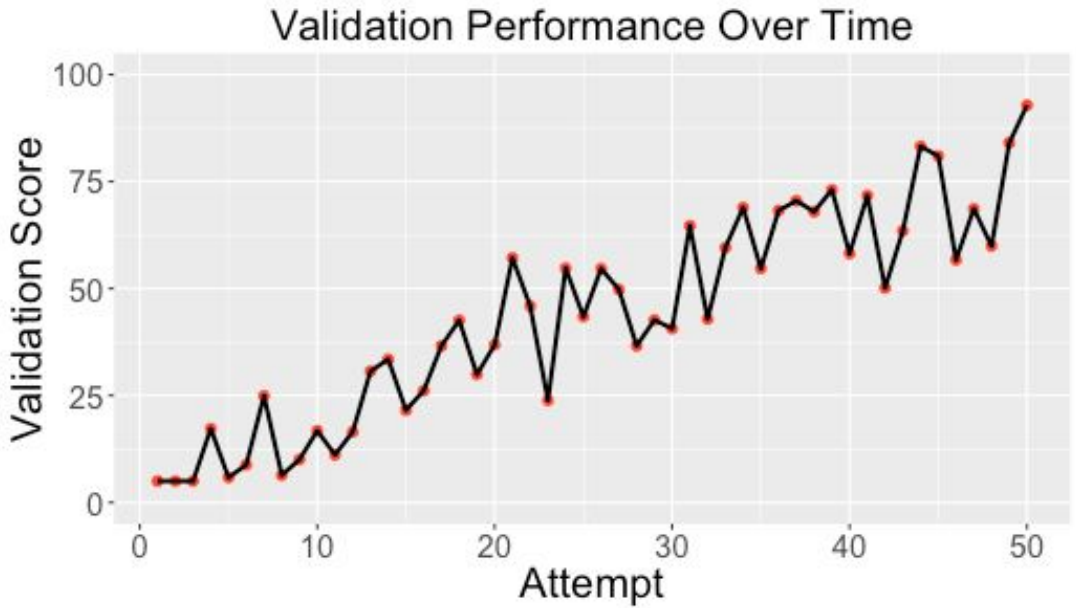
CAUTION: Entering Statistics Zone

Testing and validation
Why both?

You now have one best candidate model.

Everything looked good in validation.

Should we use it?



Testing and validation Why both?

Your validation dataset gets polluted the more you use it, so it stops being an honest measure of performance.

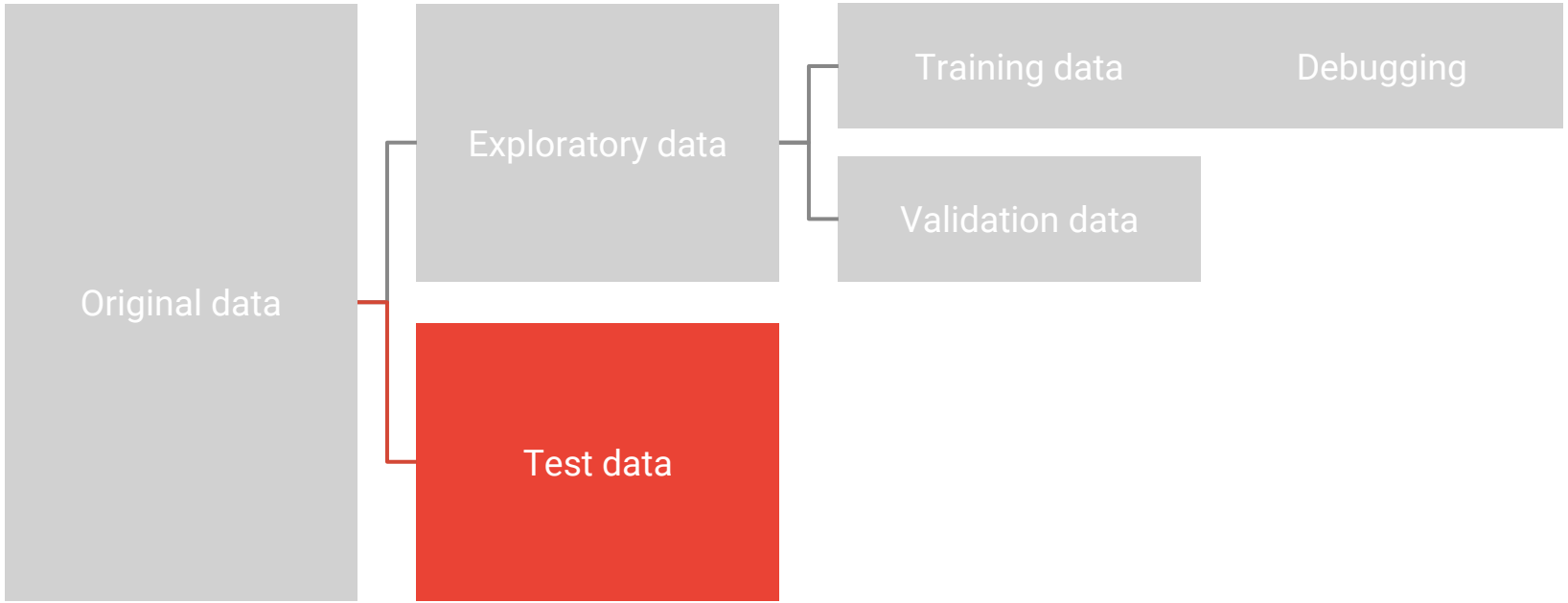
Only testing lets you know if your best is truly good enough.

Testing, the final frontier

Now entering a cleanroom with precious, completely unpolluted (new) data and estimating the model's out-of-sample performance

Image link, Author Robert Scoble

Use this, not that



The decisions

Your mission: Make the most important decision

This model is the best you have Should you kill it?

Welcome to **statistics!** This is what you need **hypothesis testing for.**

Key message

Testing is the final frontier before you take your model live

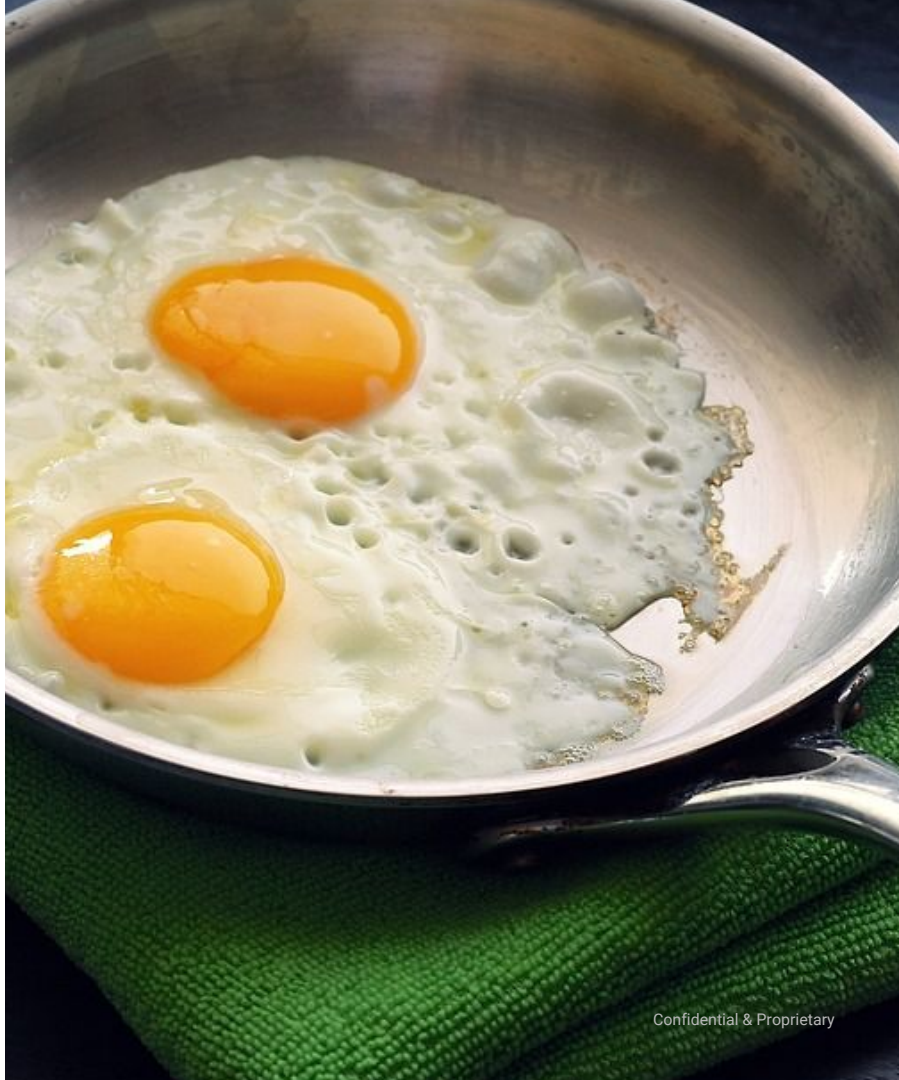
This is where statistical rigor enters the picture

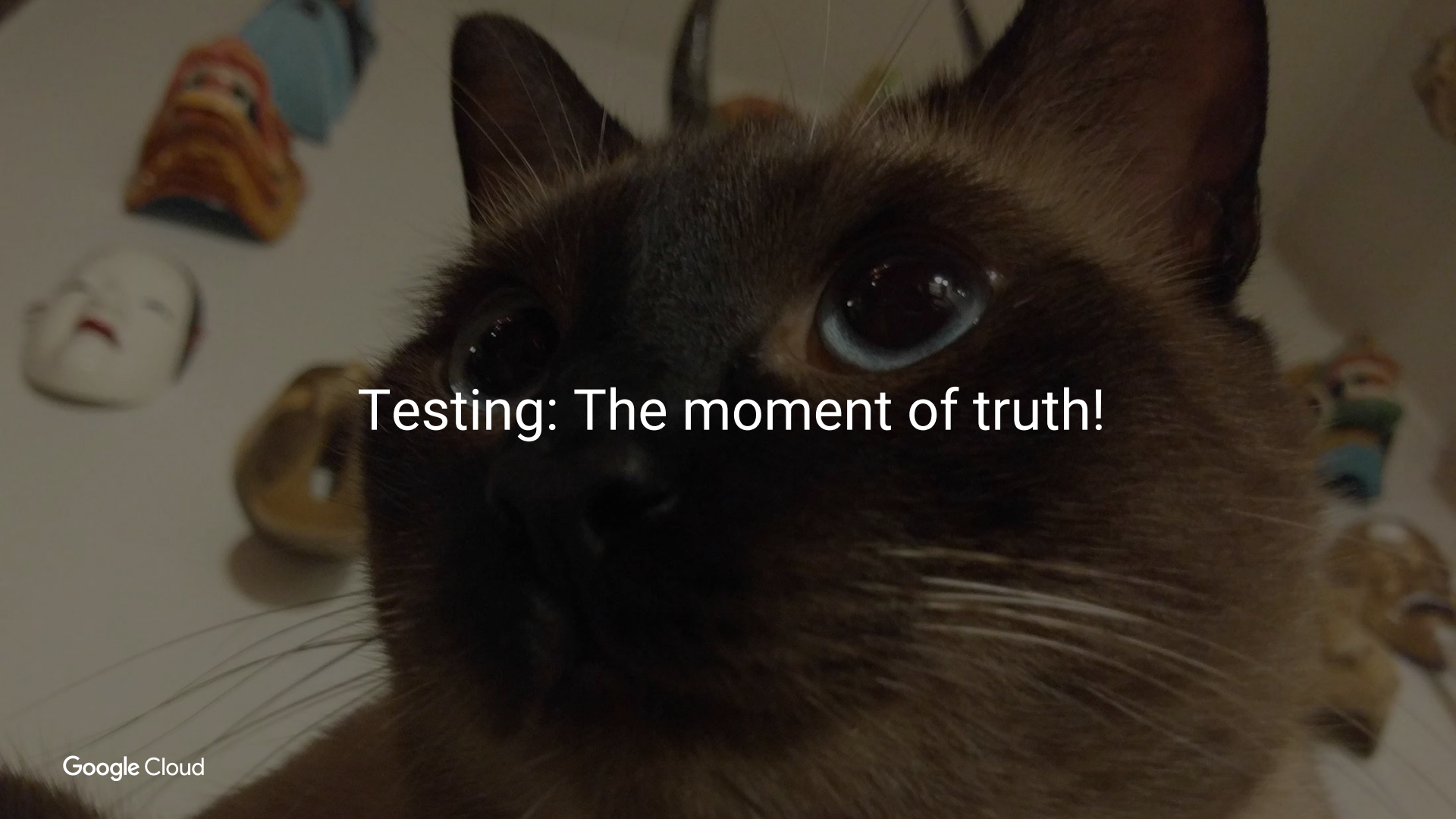
If testing fails, you can only start again if you can collect a new test dataset

In practice

There is a way to try again.

Collect a new test dataset.




Testing: The moment of truth!

Step 9 - Statistical Testing

Apply cat_finder() to the test dataset ONE TIME ONLY. Since this is testing, we'll only look at the final performance metric (accuracy) and the results of the statistical hypothesis test.

In [24]:
```
# Hypothesis test we'll use:
from statsmodels.stats.proportion import proportions_ztest

# Testing setup:
SIGNIFICANCE_LEVEL = 0.05
TARGET_ACCURACY = 0.80
```

In [25]:
```
files = os.listdir(TEST_DIR)
predicted = cat_finder(TEST_DIR, model_version)
observed = get_labels(TEST_DIR)
print('\nTest accuracy is ' + str(get_accuracy(observed, predicted, roundoff=4)))
```

```
INFO:tensorflow:Restoring parameters from ../../user/data/output_cnn_big/model.ckpt-3000
1000 predictions completed (out of 15748)...
2000 predictions completed (out of 15748)...
3000 predictions completed (out of 15748)...
4000 predictions completed (out of 15748)...
5000 predictions completed (out of 15748)...
6000 predictions completed (out of 15748)...
7000 predictions completed (out of 15748)...
8000 predictions completed (out of 15748)...
9000 predictions completed (out of 15748)...
10000 predictions completed (out of 15748)...
11000 predictions completed (out of 15748)...
12000 predictions completed (out of 15748)...
```

```
5000 predictions completed (out of 15748)...
6000 predictions completed (out of 15748)...
7000 predictions completed (out of 15748)...
8000 predictions completed (out of 15748)...
9000 predictions completed (out of 15748)...
10000 predictions completed (out of 15748)...
11000 predictions completed (out of 15748)...
12000 predictions completed (out of 15748)...
13000 predictions completed (out of 15748)...
14000 predictions completed (out of 15748)...
15000 predictions completed (out of 15748)...
15748 predictions completed (out of 15748)...

Test accuracy is 0.8416
```

In [26]:

```
# Using standard notation for a one-sided test of one population proportion:
n = len(predicted)
x = round(get_accuracy(observed, predicted, roundoff=4) * n)
p_value = proportions_ztest(count=x, nobs=n, value=TARGET_ACCURACY, alternative='larger')[1]
if p_value < SIGNIFICANCE_LEVEL:
    print('Congratulations! Your model is good enough to build. It passes testing. Awesome!')
else:
    print('Too bad.  Better luck next project.  To try again, you need a pristine test dataset.')
```

```
Congratulations! Your model is good enough to build. It passes testing. Awesome!
```

Danger! Pitfall alert

Never test on data that was involved in any way in training or validation!

Otherwise, you **will trick yourself into launching something that doesn't work.**

Step 9 is finished | You have a document clearly detailing:

- Decision process
- Whether to go live
- Performance estimate

Chief actors:

Analyst

Decision maker

Step 9 is finished | You have a document clearly detailing:

Notebook for Steps 1 and 9
Yes!
84% accuracy

Chief actors:

Analyst

Decision maker

Google Cloud

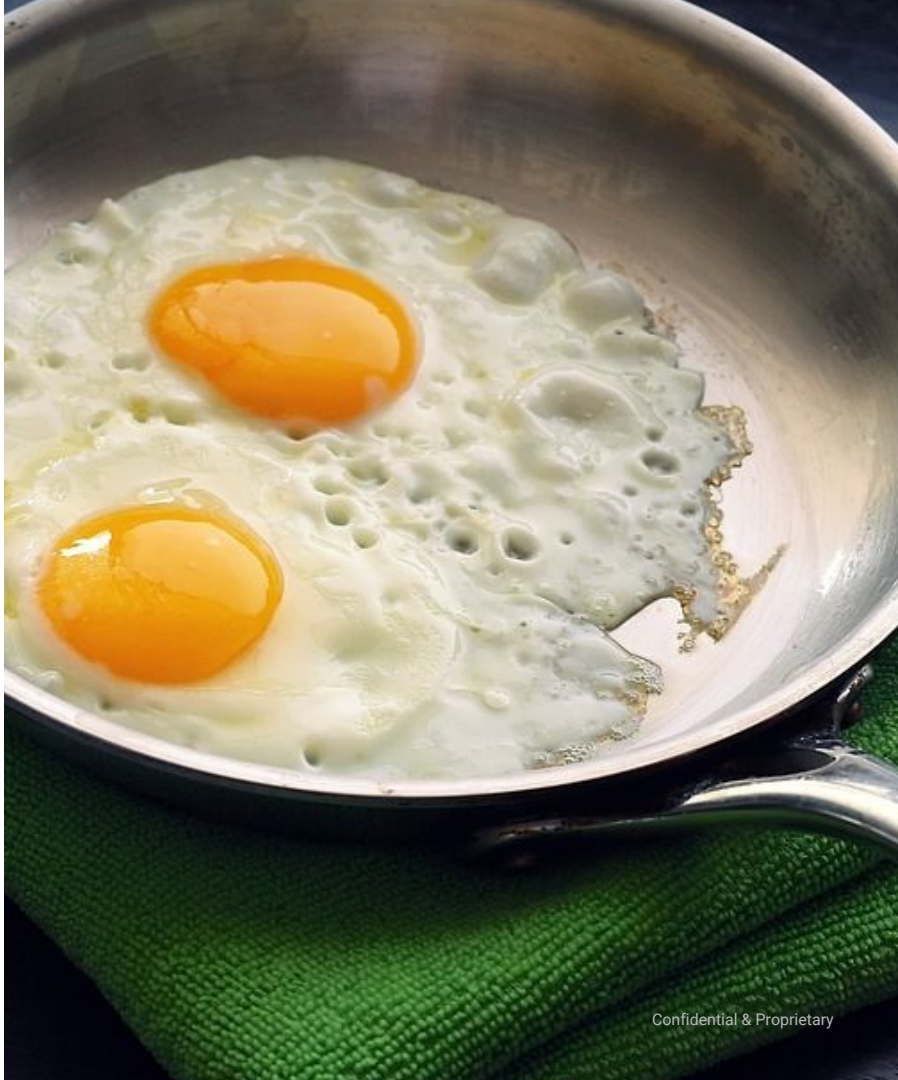
End of demo!

If you found it useful, share the love and say hi on twitter.com/quaesita.

Cassie Kozyrkov
Chief Decision Scientist, Google Cloud
kozyr@google.com; @quaesita

Google Cloud

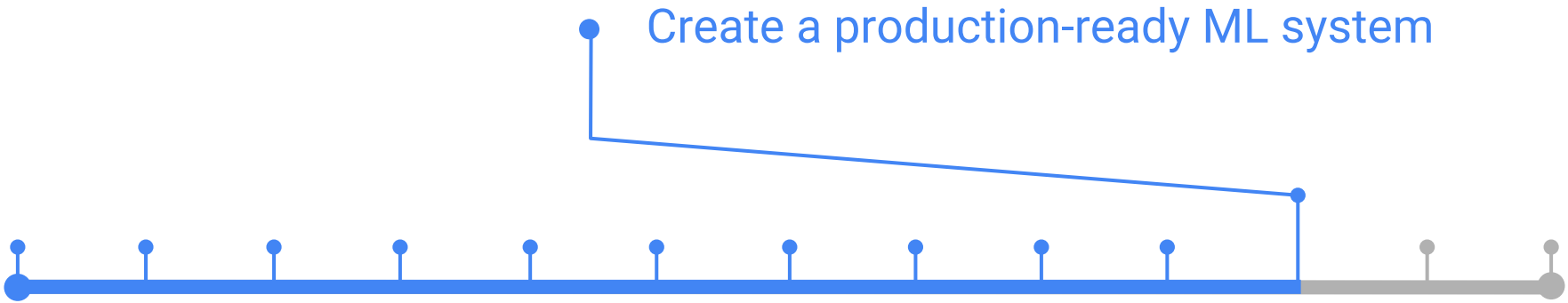
In practice



ML doesn’t only have to involve big launches in industry production systems. Use it at home too.

For you, a “launch” might refer to a decision to rely on some model from now on.

Step 10 | Build your ML system



Chief actors:

Engineer

You can find a tutorial on Step 10 at:

github.com/google-aai/tf-serving-k8s-tutorial

Authors: Brian Foo and Ron Bodkin

Brian
Foo

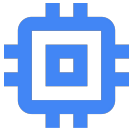
Ron
Bodkin

Google Cloud

Step 10 is finished | Your engineering effort got you:

- Production-ready system
- Automated retraining
- Safety nets

Chief actors:



Engineer

Step 11 | Make launch decision

Test your system in its live environment

Chief actors:

Analyst

Decision maker

Training-serving skew

Beware the training-serving skew

To ensure your model works where you'll be serving it, test again in **production**!

Danger! Pitfall alert

Don’t launch all at once.

It’s better to look before you leap, because what if your users change their behavior in an unexpected way as a result of your launch?

Plan a gradual ramp-up where you start serving the model to a small fraction of traffic.

Live-traffic experiments are a great idea!

Live traffic experiments

Second time we'll need **statistics** for decision-making: Launch it or don't launch it?

Make this choice rigorously by running an experiment.

Live traffic experiments

Components of a real experiment:

1. Hypothesis
2. Different treatments
3. Randomization to treatments

Live traffic experiments

You can do real experiments here

1. Performance good enough?
2. ML system vs no ML system
3. Live traffic sent at random to ML system or old system*

*When you're dealing with massive scale, a good idea is to do a 1% live traffic experiment. (1% your ML system, 99% unchanged)

Step 11 is finished | You've ramped up and gradually you're:

- Serving model to users
- Using a policy layer
- Staying cautious

Chief actors:

Analyst

Decision maker

Step 12 | Monitor and maintain

Does your model stand the test of time?

Chief actors:

Reliability engineer

Analyst


But it worked when
I bought it

Danger! Pitfall alert

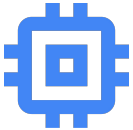
Don’t forget to do performance checks.

Monitoring and maintenance plans for ML systems help keep them working reliably.

Step 12 is never finished | But a good start is having:

- Monitoring plan
- Maintenance plan
- Tracking dashboards
- Good documentation

Chief actors:



Reliability engineer

Analyst

Thank you for learning!

If you found this useful, share the love and say hi on twitter.com/quaesita.

Cassie Kozyrkov
Chief Decision Scientist, Google Cloud
kozyr@google.com; @quaesita

Google Cloud