ML Test Rubric

Roman Degtiarev

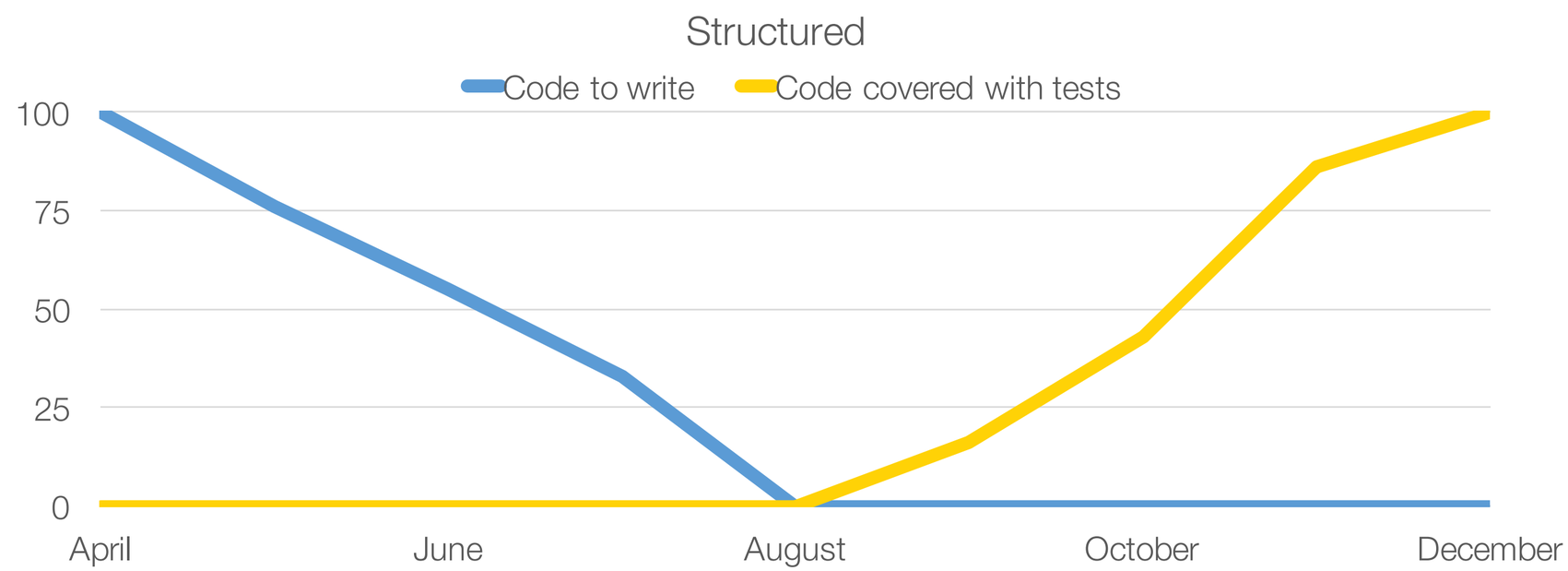
Targets for a round

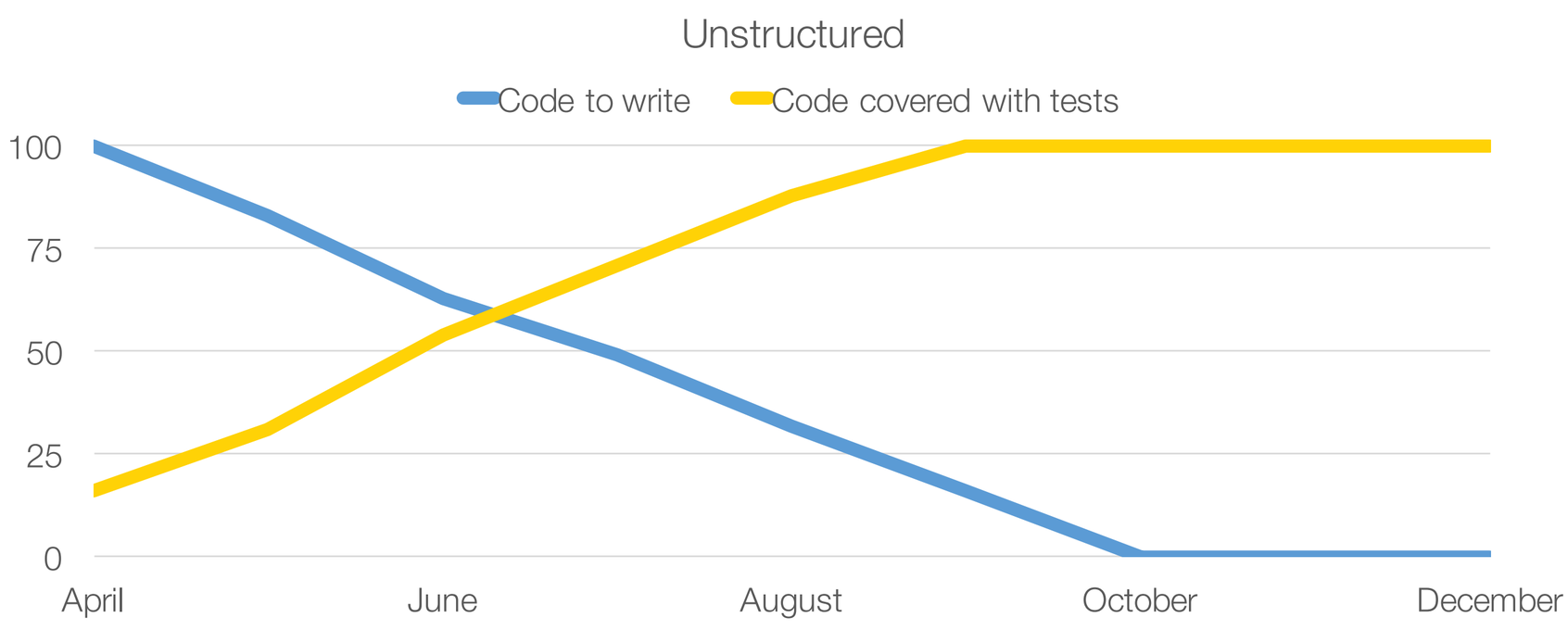
- Recap
- Why do we need tests
- Why do we need specific tests in ML
- How to score your system
- Tests field and examples
- Final score

Reproducibility and reliability

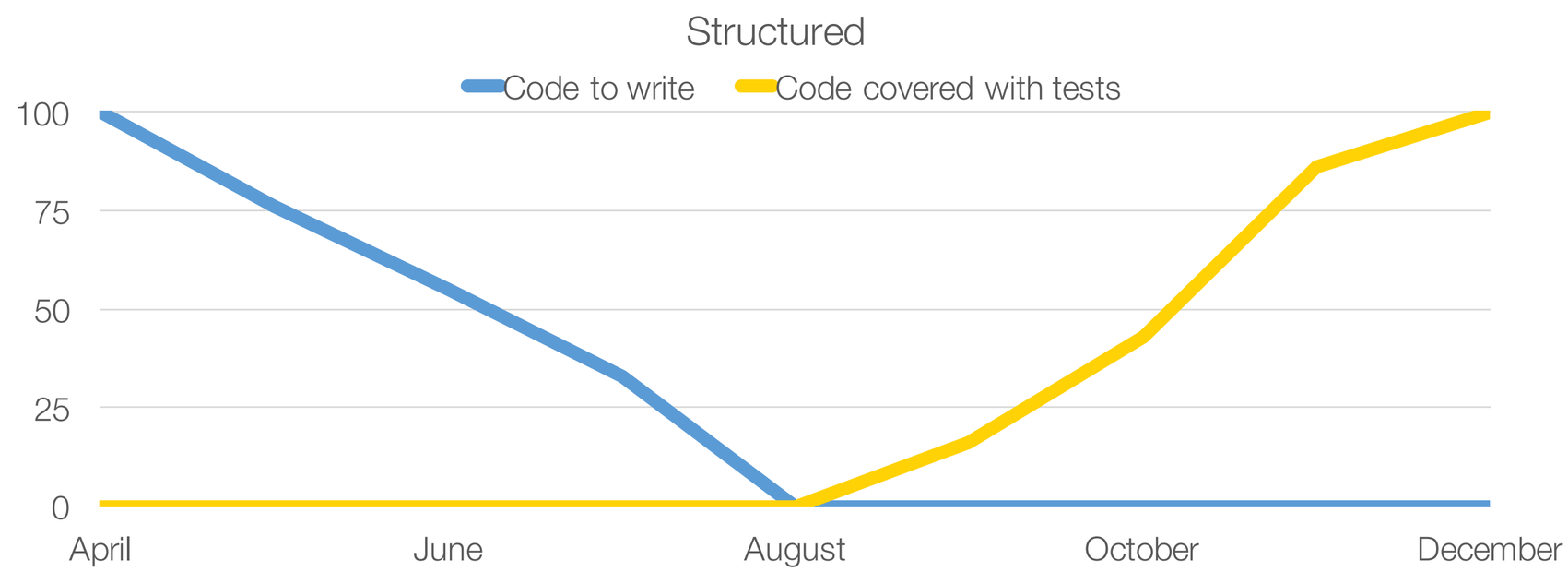
- Reproducibility means that you will get same result each time (*somehow or other*)
- Reproducibility means that anybody could verify your result
- Reliability means that you know what's going on and able to control each step

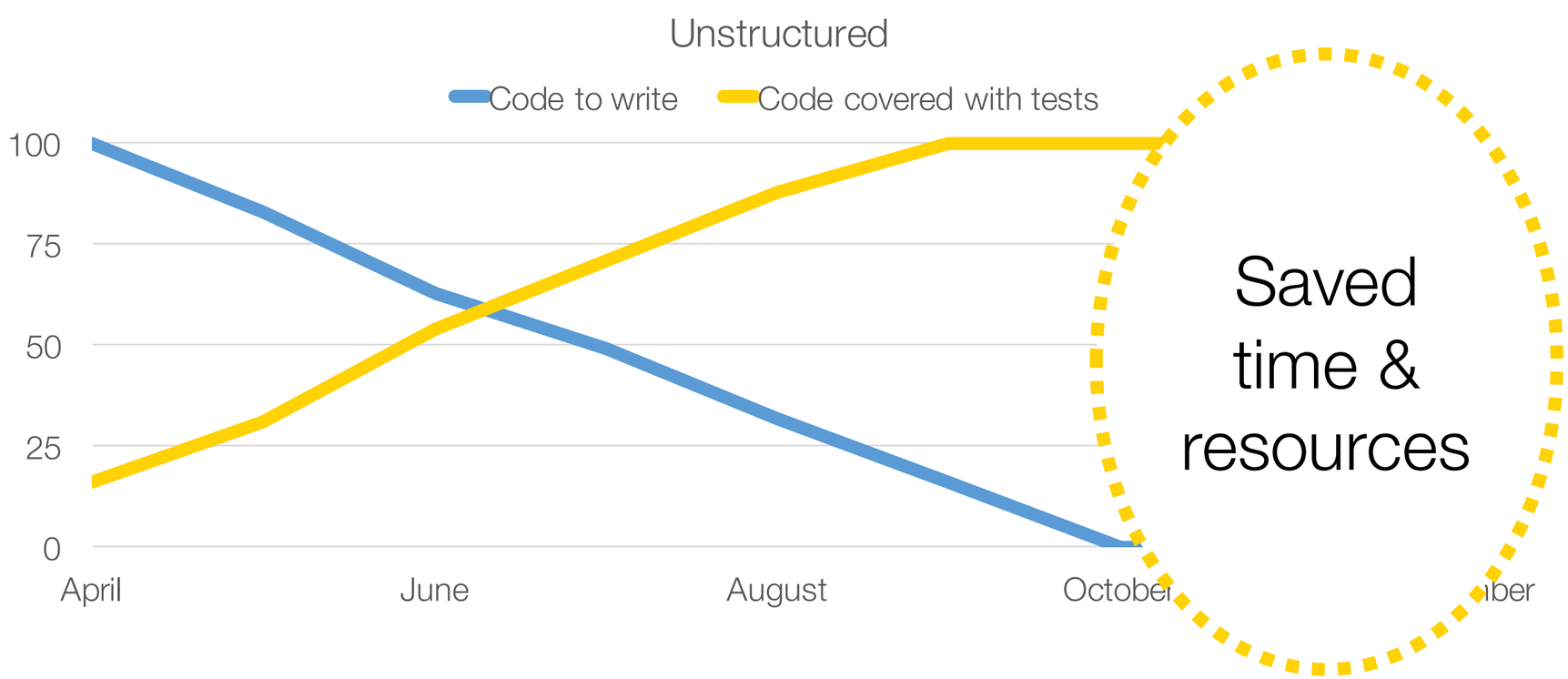
Model based testing





Model based testing





Main properties*

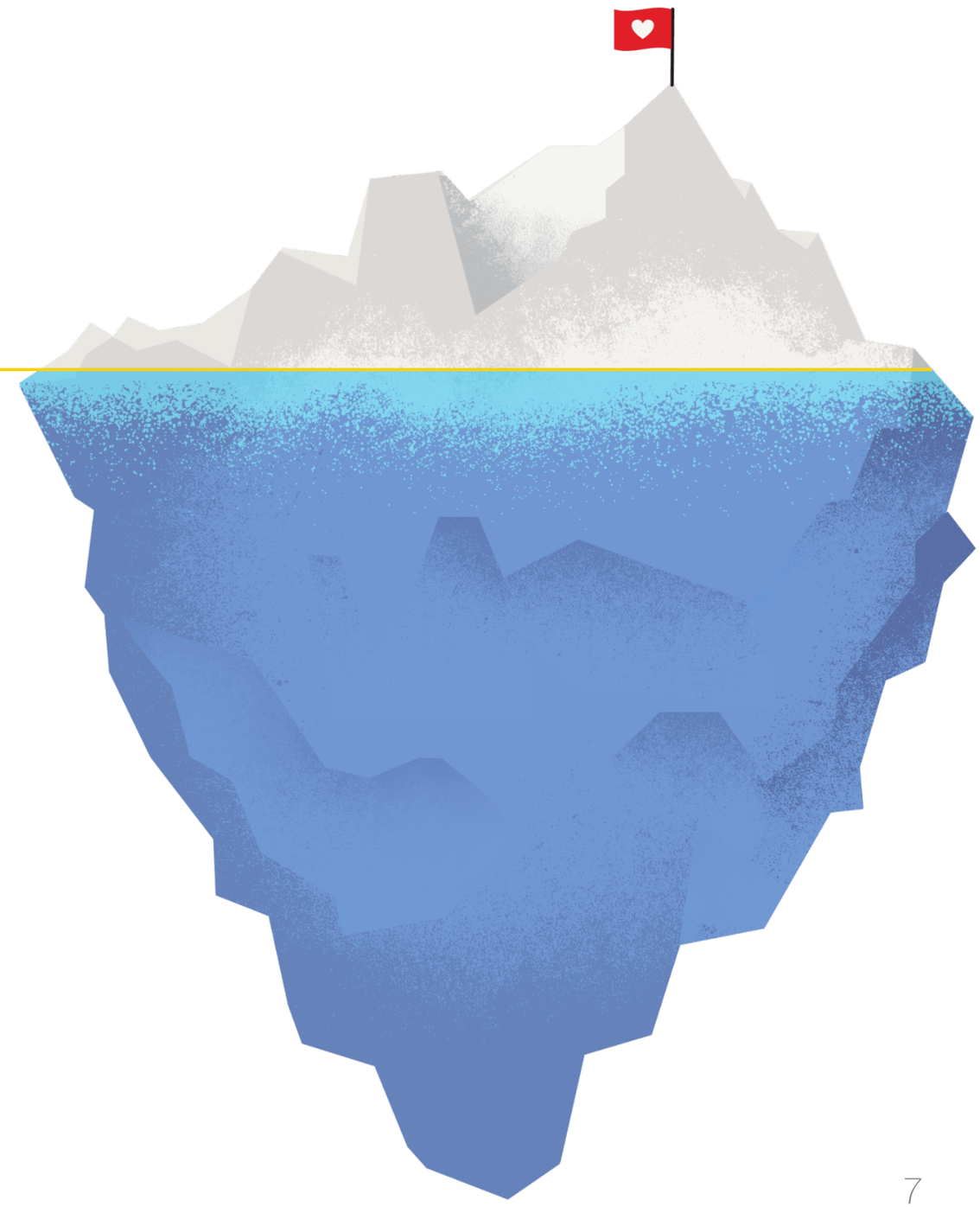
- Empirical risk
- Learning/serving metrics
- Learning time
- Memory consumption while learning

* as it seems to young analytic

Iceberg Rule

- Learning time
- Serving metrics

- Computation time
- Reproducibility
- Staleness
- Privacy



Deep monsters

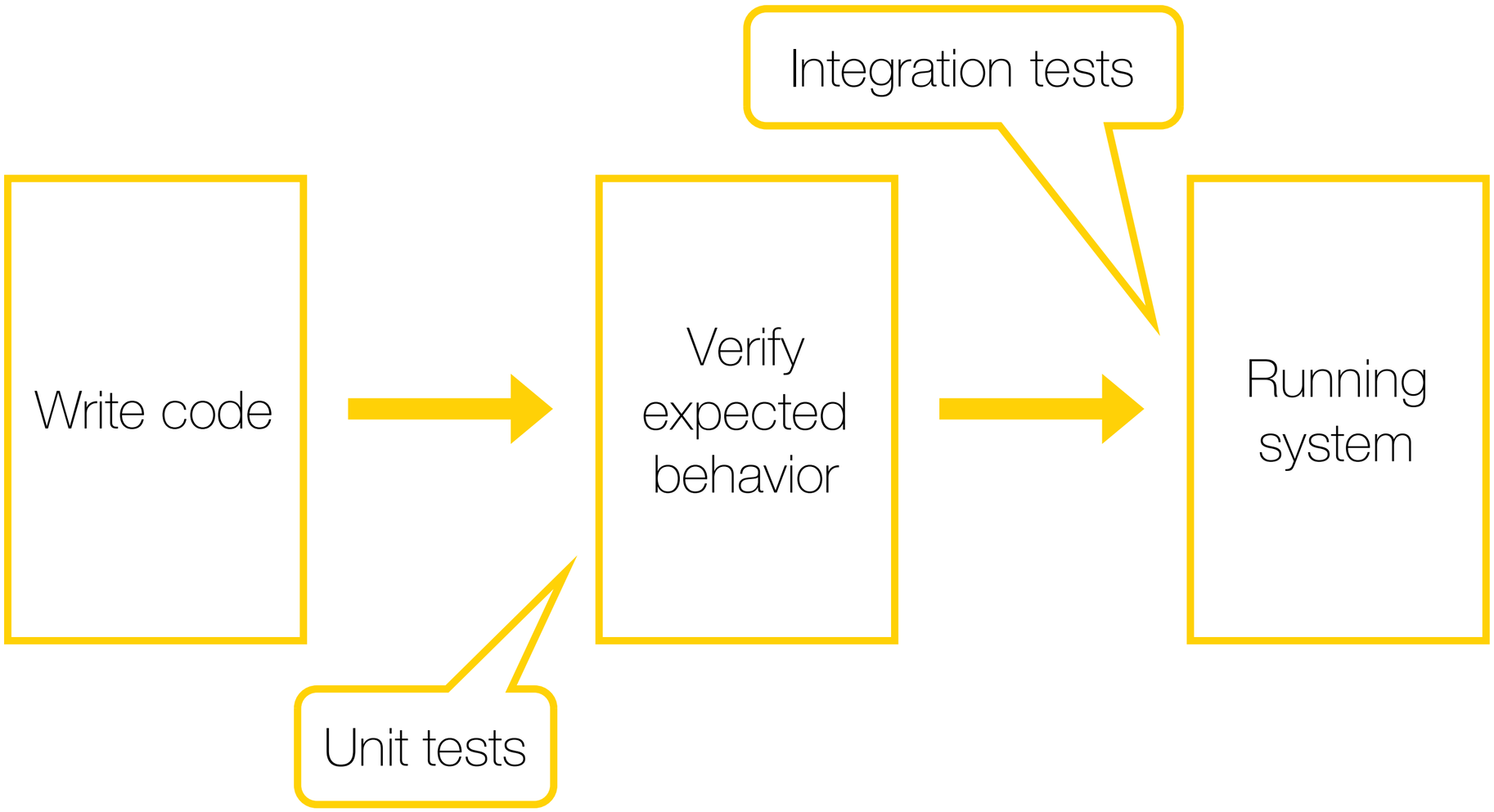
- Domain-specific problems
- Business-rules
- Integration
- Rollbacks


ADVENTURE
TIME

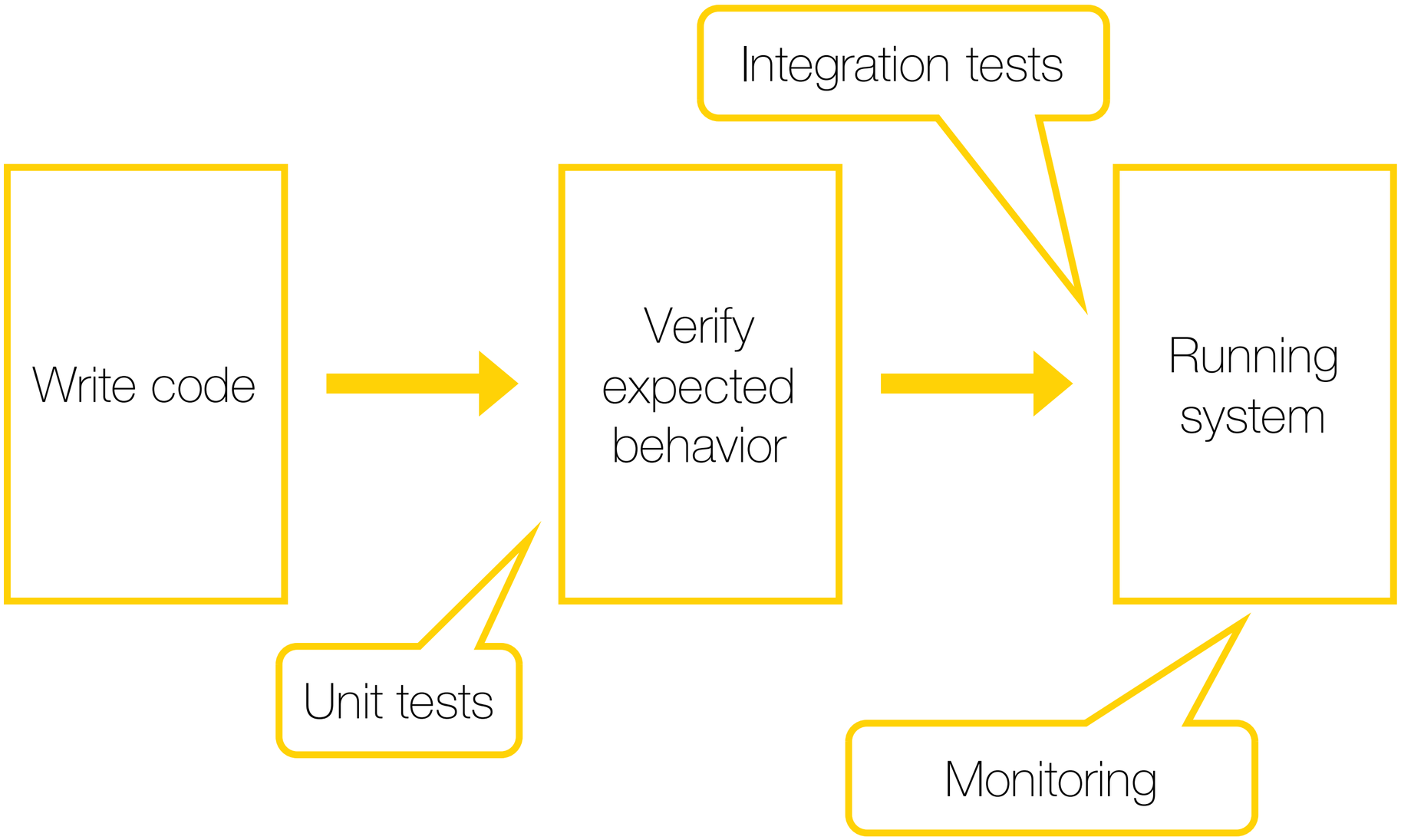
Testing code

Testing code

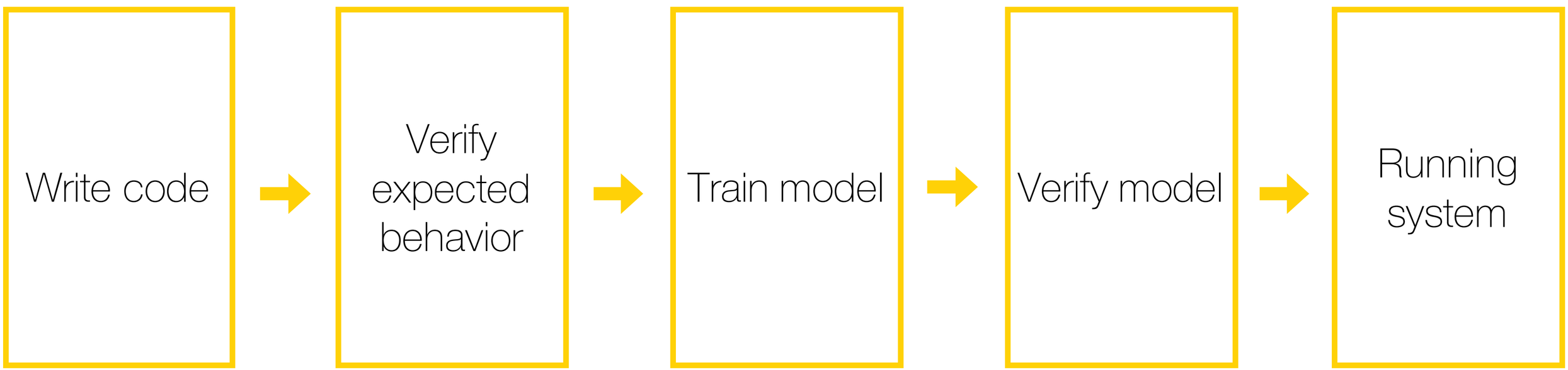
Testing code



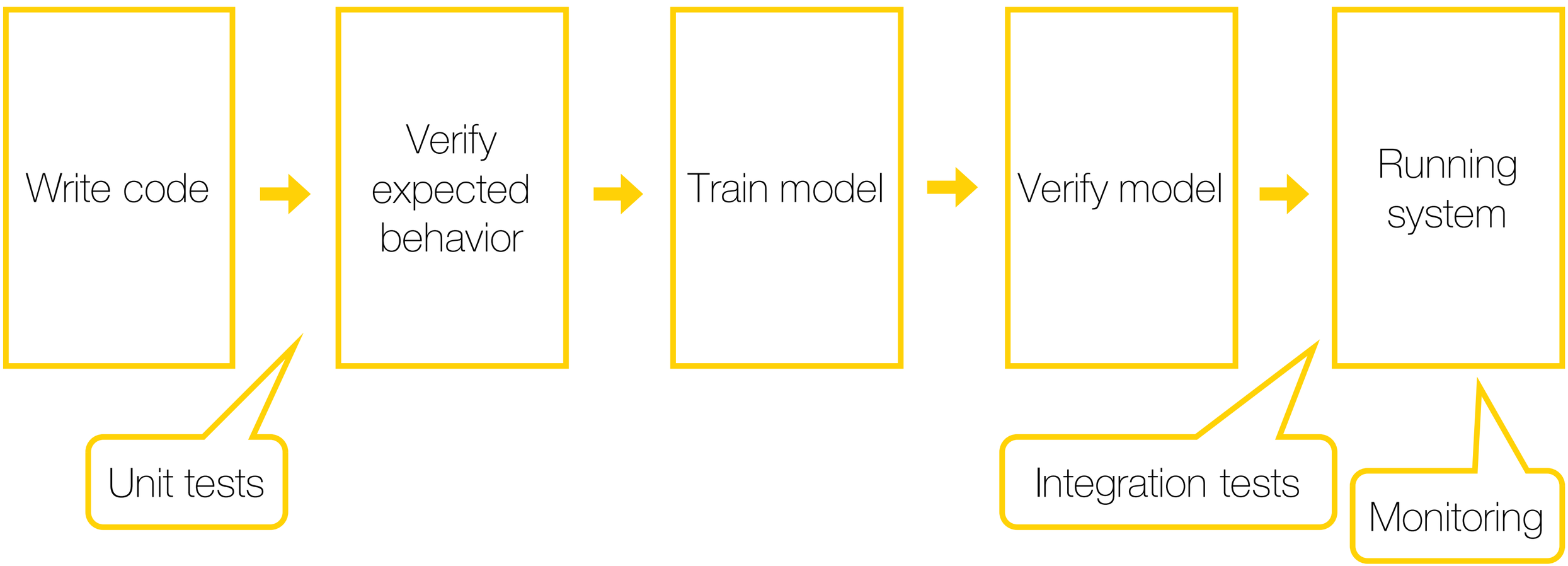
Testing code



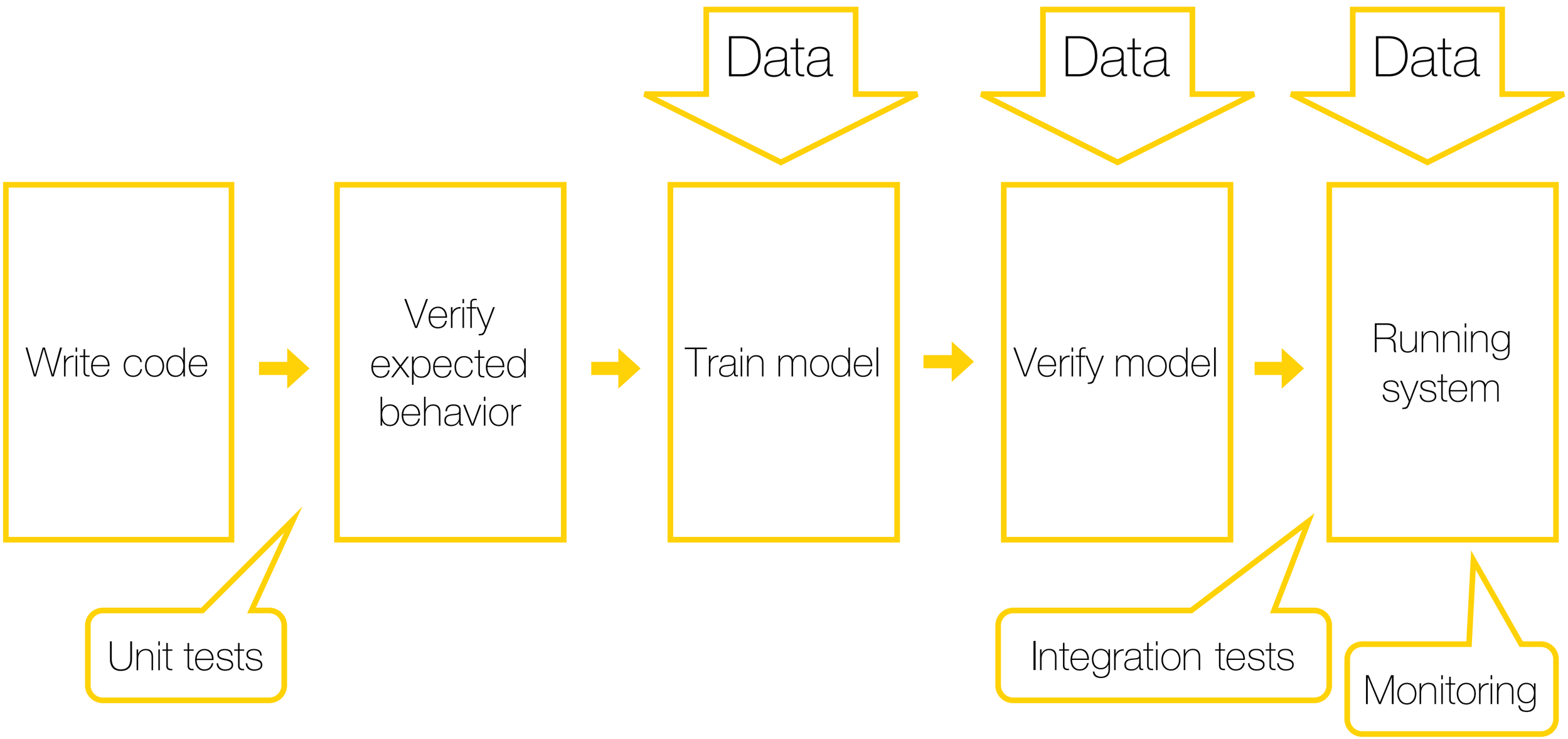
Testing ML code



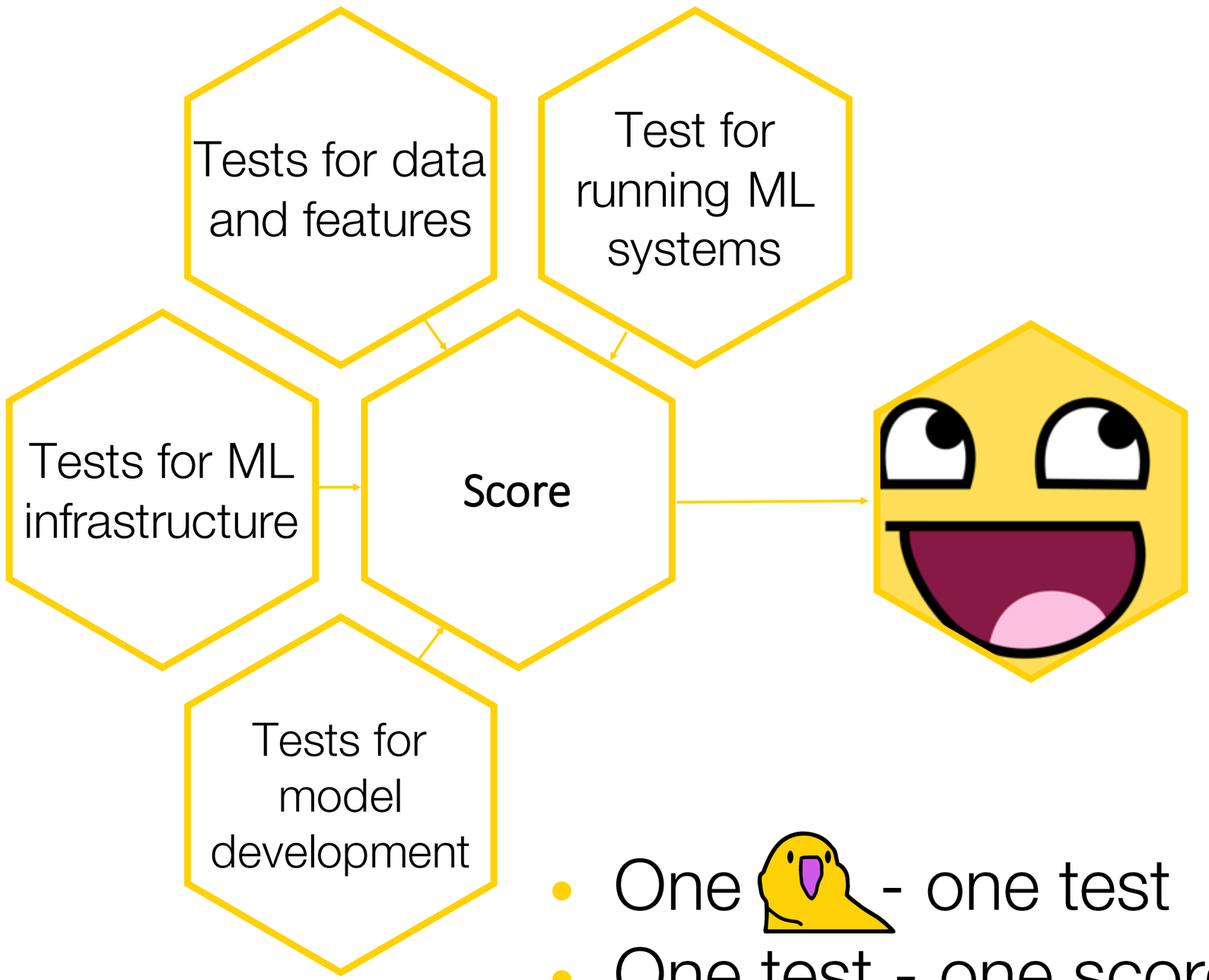
Testing ML code



Testing ML code



An ML Test Rubric

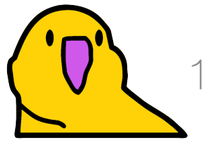


- One - one test
- One test - one score point

Tests for data and features

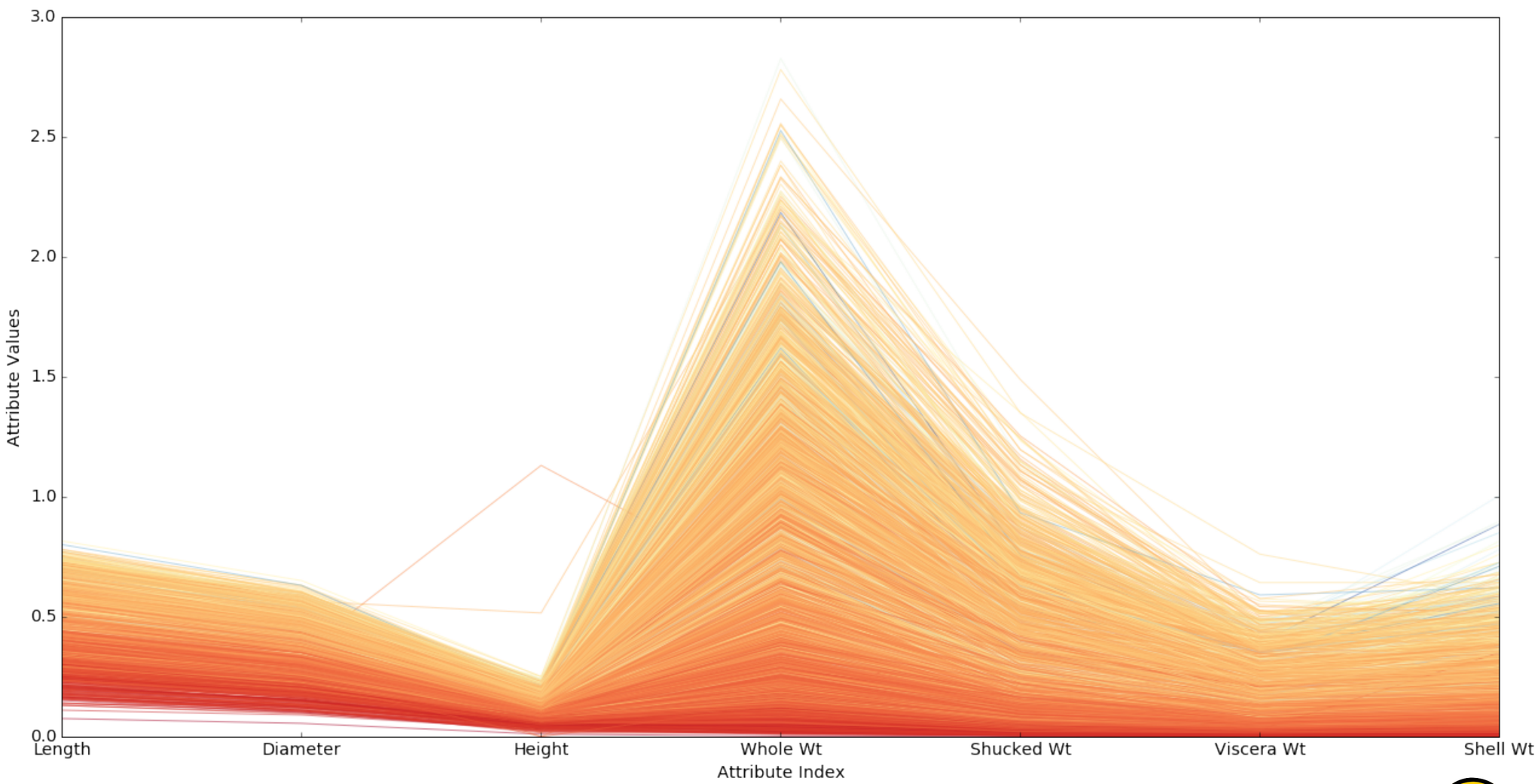
Feature creation code test

- Unit-test for code (mandatory!)
- Third-party code tests
- Third-party service test
- **Redis** under load: must be str, should be JSON -> break model

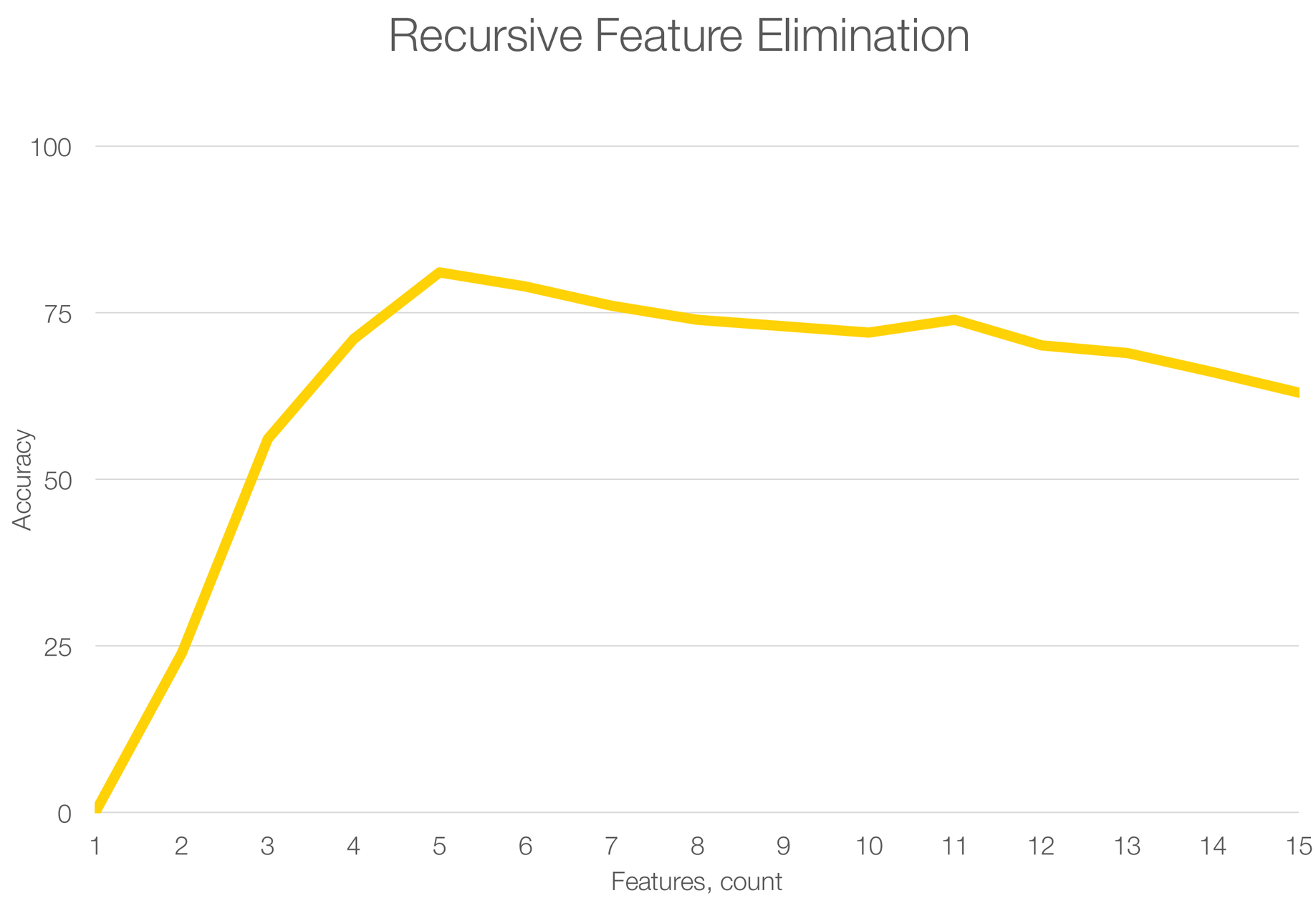


Distribution test

- Test that distributions match your expectations

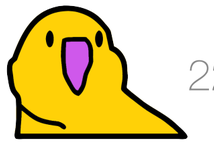


Feature importance test

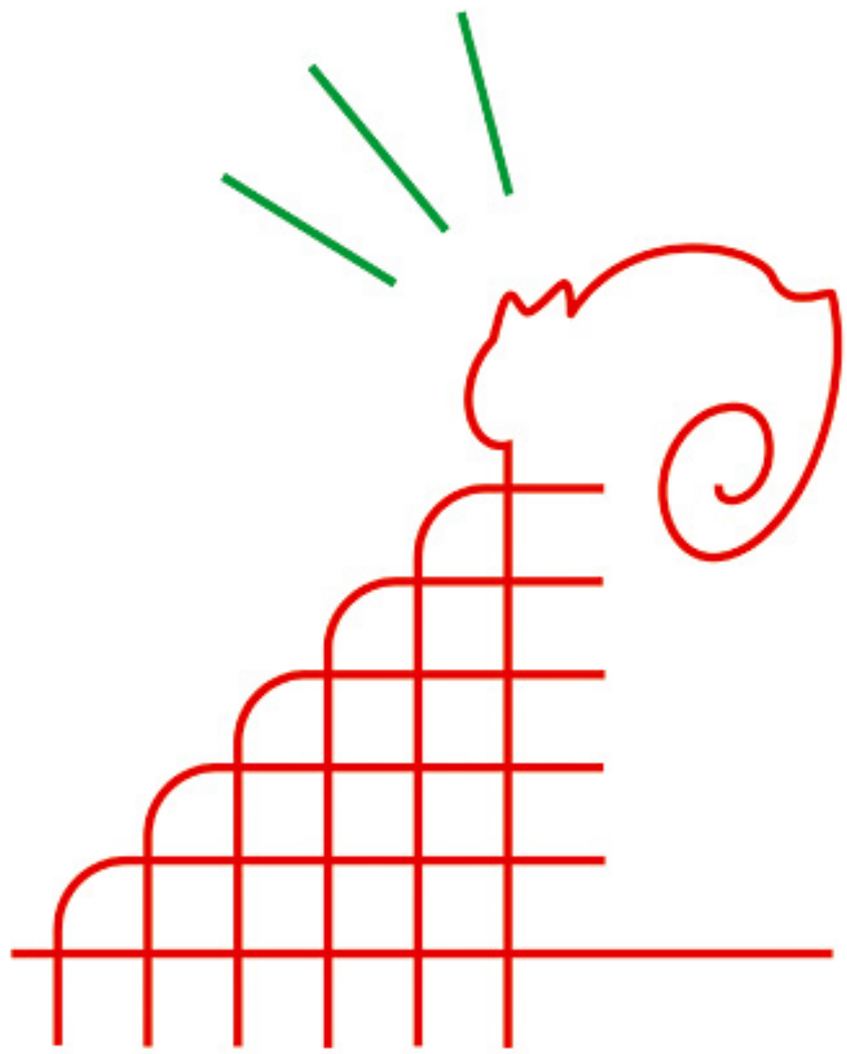


Feature cost test

- Added inference latency (CPU/GPU usage)
- RAM usage
- Upstream data (additional traffic)
- Additional instability



Unusable features test



Exclude **manually** marked as unusable by expert

Privacy test

- Privacy leak through data
- Privacy leak through results
- Privacy leak through pipeline

Extension time test

- What time do you need to add new feature to the model?
- What time do you need to retrain model?
- What time do you need for any further improvement?

Tests for model development

Version Control Test

- Do you fix/commit your results somewhere?
- Do you cover all the components of your system?
- How easy is it to rollback?
- Do you store binary models somewhere?

Offline vs online metrics

- Is your empirical risk minimization correct?
- What is correlation between backtest metrics and production?
- Moreover: do you achieve any gain in business?

Hyperparameter tuning

- Test impact of each hypermarameter (hey, EC2)
- GridSearch/Hyperopt
- Data parallelism

Staleness test

- How long could model be served without updating?
- Where is tradeoff between model properties and update costs?

Baseline test

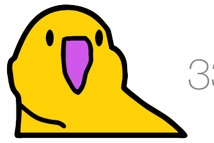
- Create as simple model as you could
- Use mean/previous/etc.
- Use random number generator!
- Use out-of-the-box models

Test on important data slice

- Select important data slices
- Try to make slices as varied as possible
- Measure metrics on each slice
- Extreme speedup in development
- Help to avoid performance issues masked by global summary metric

Implicit bias test

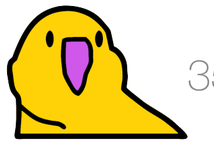
- Also could be implicit-association test
- Isn’t your model prejudiced?
- Is your data diverse enough?
- Overfit on business rules or human labeling results!



Tests for ML infrastructure

Test on small data slice

- Extreme speedup on development
- Avoid preprocessing data on each run
- Danger: bias on slice!



Reproducibility test

- Use seed
- Use seed for seed
- Use stable data slices
- Train twice on same data with same seed
- Train twice with different seed on same data
- Help to avoid lucky strike on initialization and learning instability

Unit-test on specification

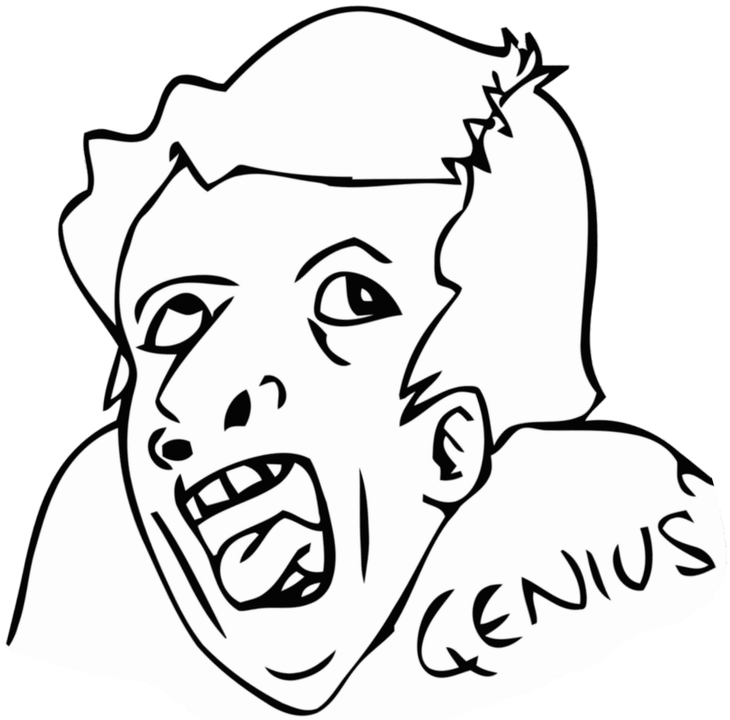
- Parse config
- Create checkpoint
- Check checkpoint for restore
- Useful assertions (metrics, time, etc)
- …and **raise errors** all the time

Integration test

- Reproduce pipeline to catch errors
- Remember about test on small data slice (!)

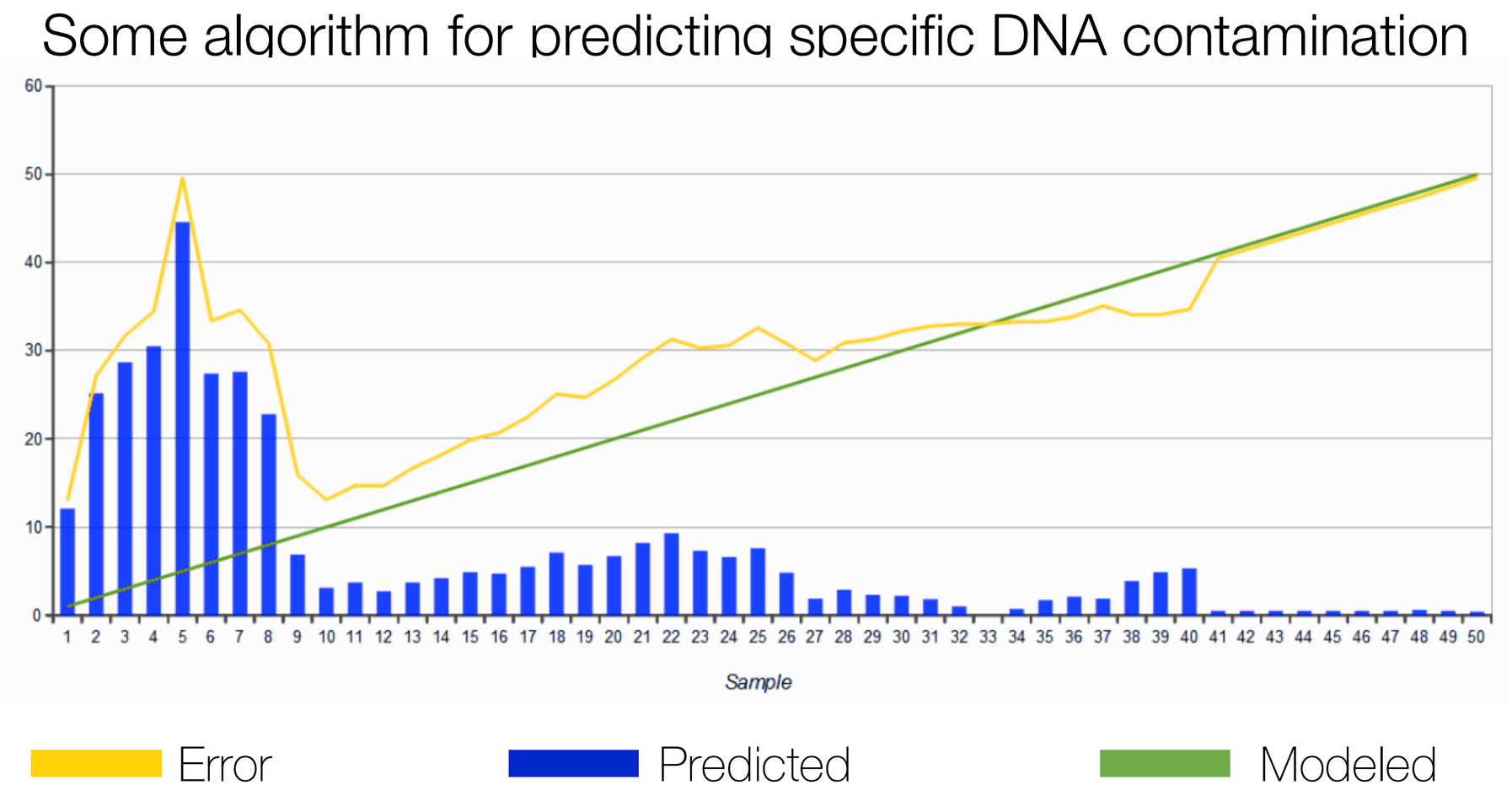
Test model quality

- Understand what do you need to measure
- You always could compute metric, but it could not be useful (much wow)
- Remember about measuring metrics on important slices



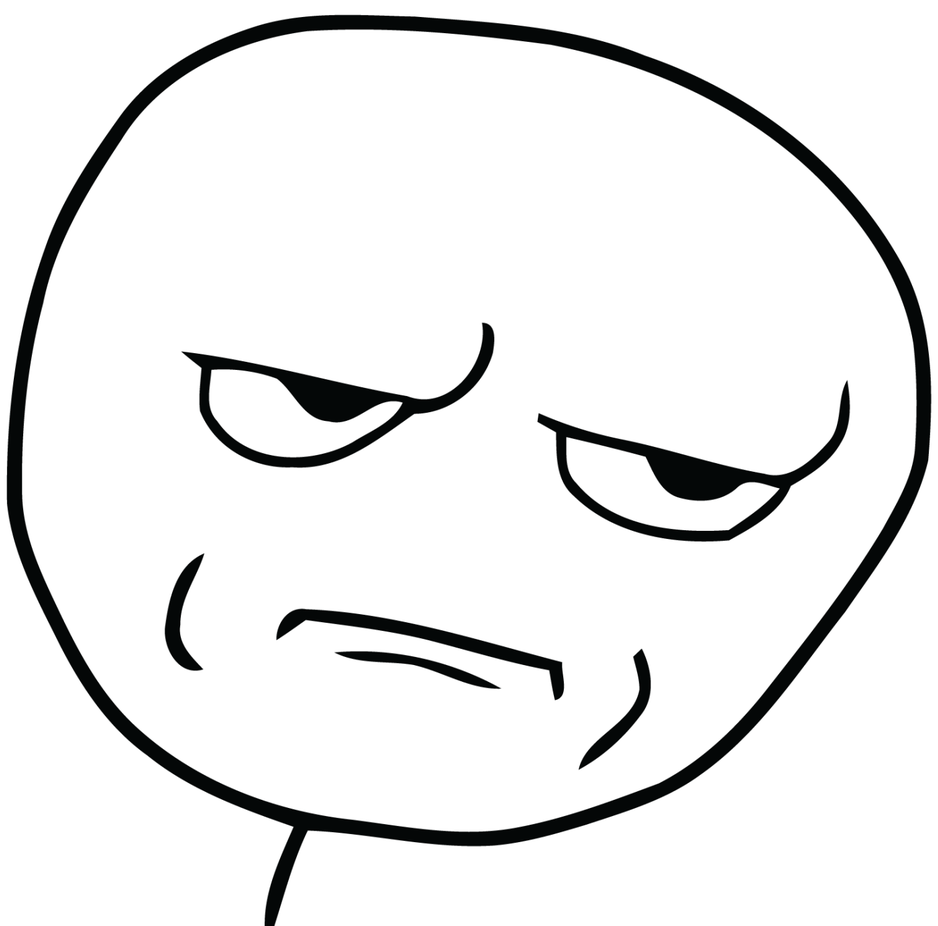
Canary test

- Test model on synthetic data
- Test that model will fail in known cases



Rollback test

- Test existence of backup
- Test ability to rollback from backup
- Test that everything works fine after rollback



Monitoring Tests for ML

Upstream instability test

- What if data provider would go down?
- What if several servers turn off?
- What if database would be overloaded?

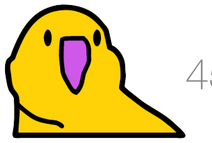
Data invariants test

- Check that stream of data is consistent – ranges, frequency, balance, conditions, etc.

- Should feature A always be in range [x, y]?
- Should class distribution always be like 10:1?
- Should sum of features [A, B, C] always equal 1?

Training/serving skew test

- Features in train and serve are always same order/value?
- Tradeoffs in serving computation code

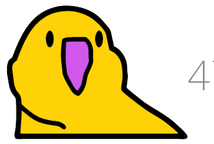


Test for model staleness

- Real-time staleness (bias vs. error)
- Retrainable model - when it’s time to change methodology?

NaN / Inf test

- Could you get NaN / Inf on train/serve?
- What should you do with them?
- Is it critical or not (raise alert)?



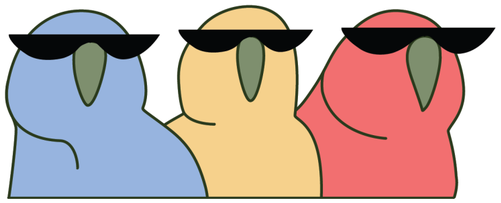
Leak in resource usage

- Not like “memory leak” – it’s about computation complexity
- Could be dramatic or slow-leak
- Training speed
- Serving latency
- Throughput
- RAM usage

Regression in prediction quality

- Non-zero bias
- Canary process
- Help to monitor for real world changes

Scoring results



- 0 points: More of a research project than a productionized system.
- 1-2 points: Not totally untested, but it is worth considering the possibility of serious holes in reliability.
- 3-4 points: There's been first pass at basic productionization, but additional investment may be needed.
- 5-6 points: Reasonably tested, but it's possible that more of those tests and procedures may be automated.
- 7-10 points: Strong levels of automated testing and monitoring, appropriate for mission- critical systems.
- 12+ points: Exceptional levels of automated testing and monitoring.

Additional materials

- What's your ML test score? A rubric for ML production systems [Eric Breck, Shanqing Cai, Eric Nielsen, Michael Salib and D. Sculley, Google Inc., 2016]

Thank you!