

Help!

(Tools Metrics can make)

Kyle Lahnakoski
Engineering Productivity, Mozilla

ActiveData

ActiveData stores JSON documents

```
{
  "run": {
    "timestamp": 1448535906
  },
  "result": {
    "test": "test_getAll.js",
    "duration": 3.016
  }
}
```

```
{
  "run.timestamp": 1448535906,
  "result.test": "test_getAll.js",
  "result.duration": 3.016
}
```

ElasticSearch does the heavy lifting as an in-memory columnar datastore

ActiveData

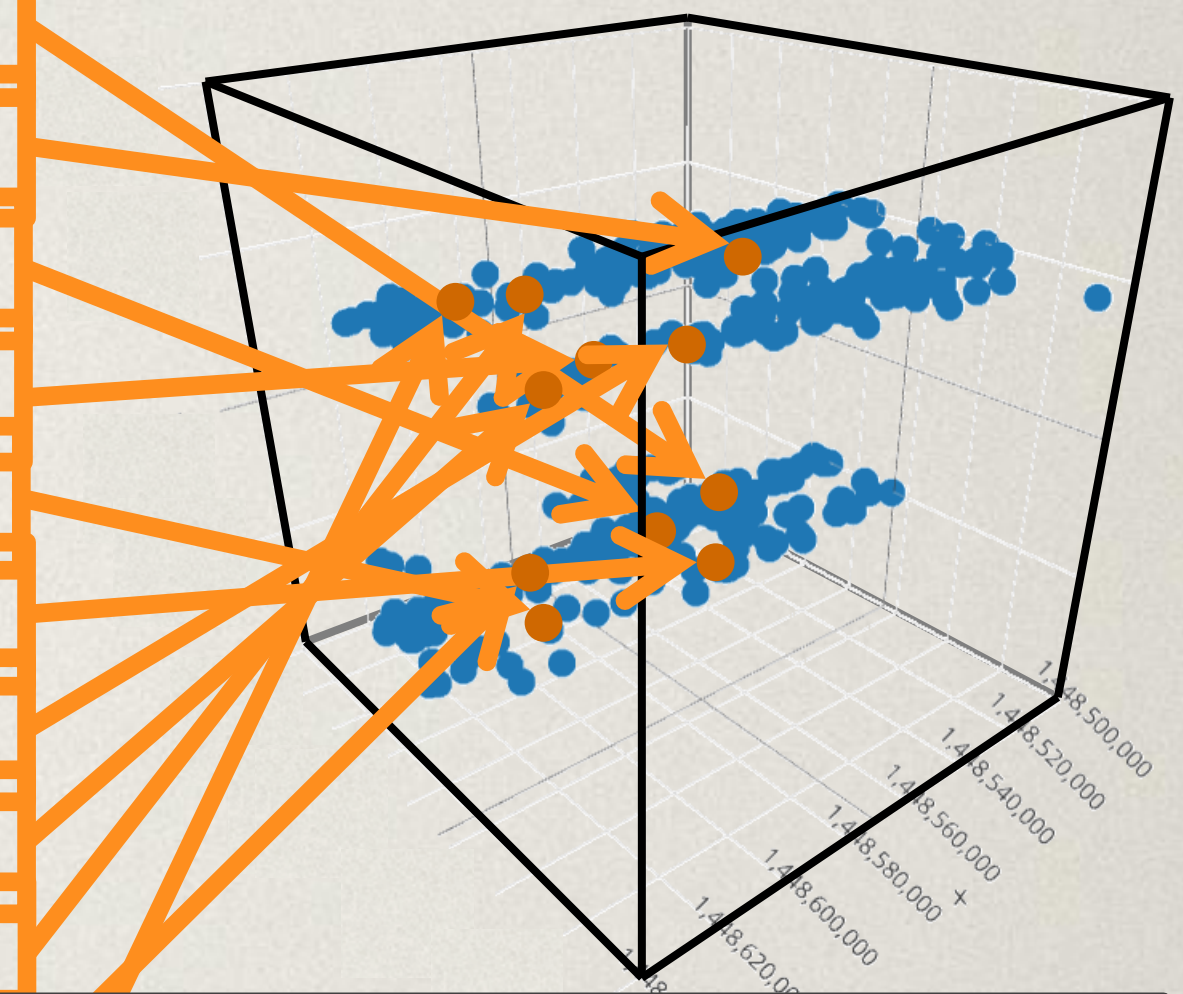
result. duration	result. test	run. timestamp
3.016	test_getAll.js	1448535906
1.601	test_660156.js	1448536787
1.718	test_getAll.js	1448536787
1.780	test_660156.js	1448536787
2.127	test_getAll.js	1448536787
3.188	test_getAll.js	1448536787
2.272	test_660156.js	1448536787
2.247	test_660156.js	1448536787
1.730	test_660156.js	1448499043
2.045	test_getAll.js	1448499043
1.571	test_660156.js	1448499043

```
{  
  "run.timestamp": 1448535906,  
  "result.test": "test_getAll.js",  
  "result.duration": 3.016  
}
```

ActiveData queries
documents as if they are
records in a table

ActiveData

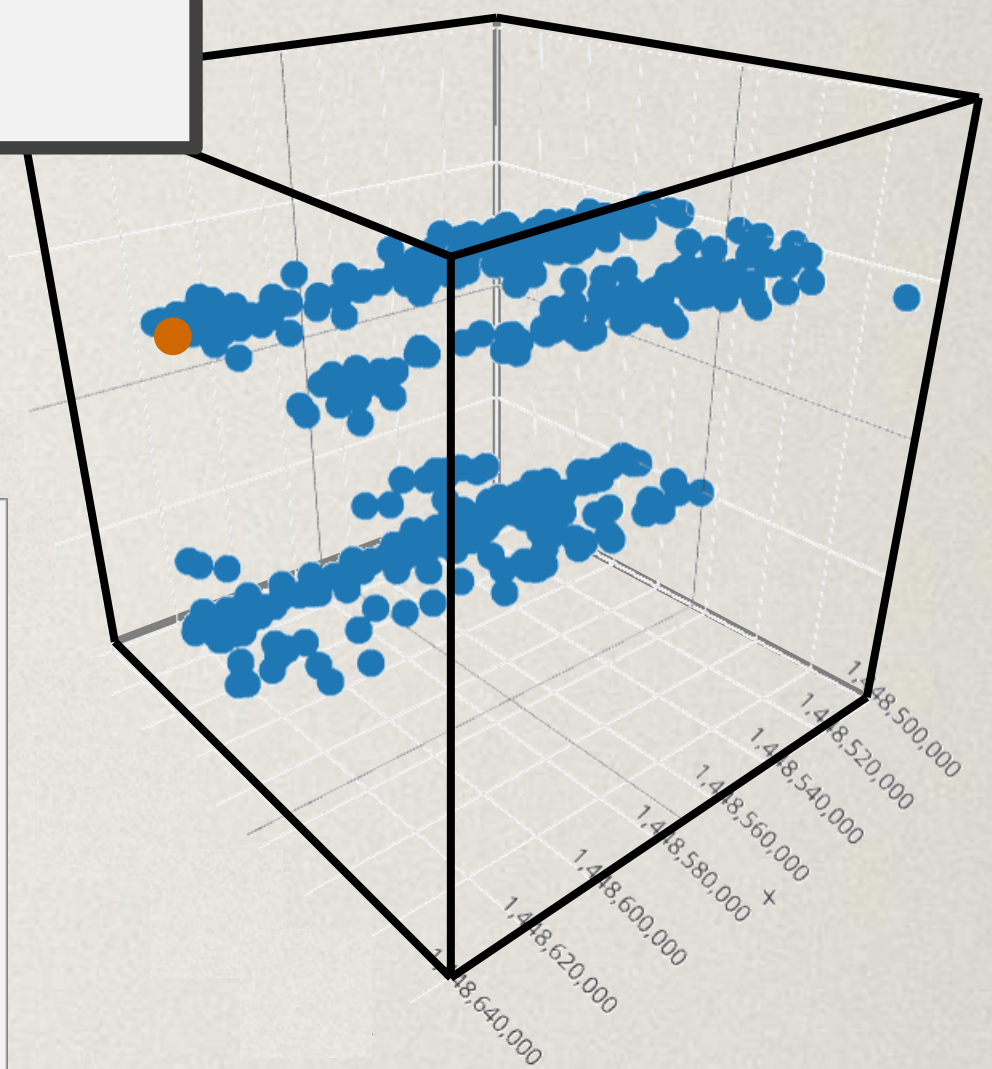
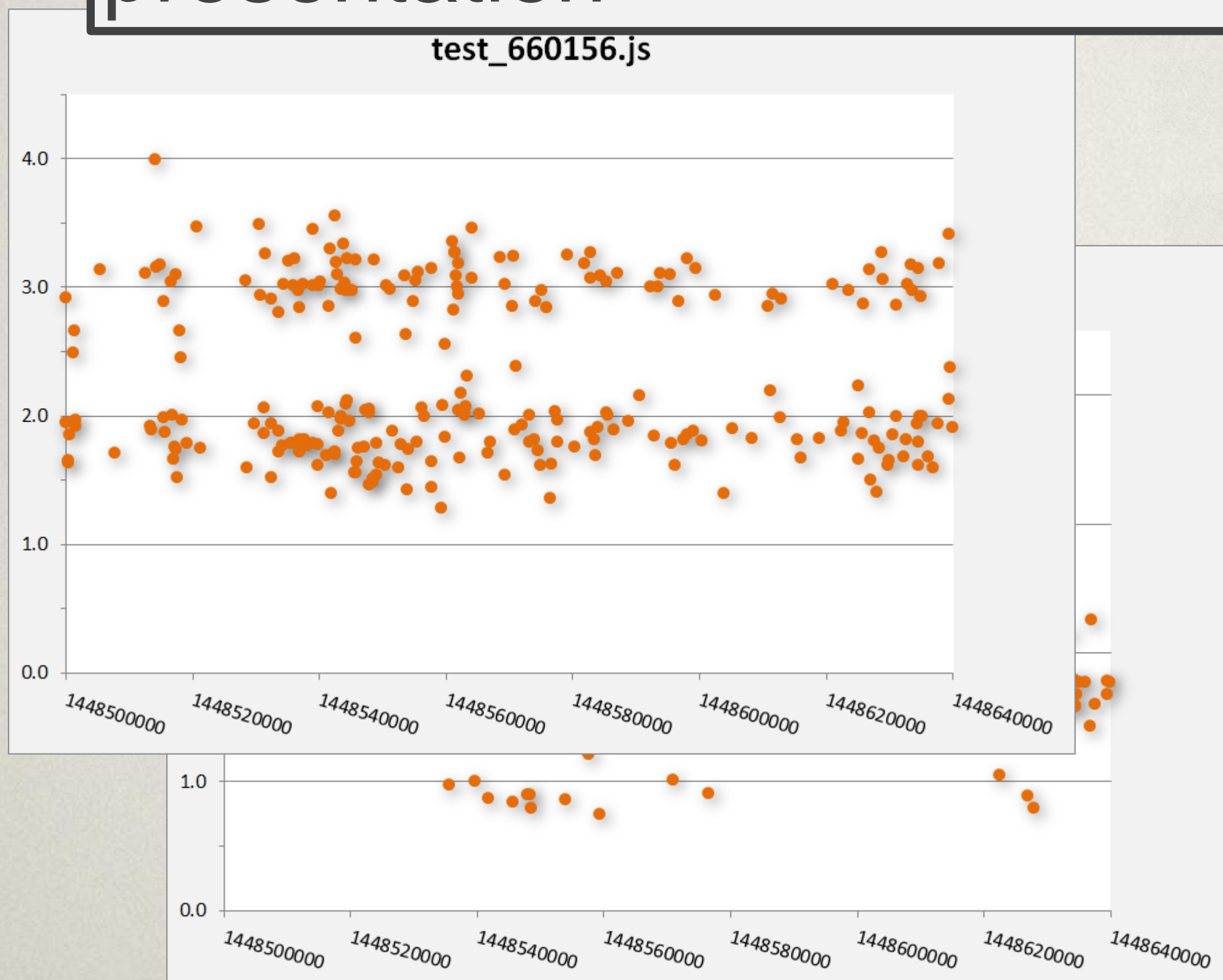
result. duration	result. test	run. timestamp
3.016	test_getAll.js	1448535906
1.601	test_660156.js	1448536787
1.718	test_getAll.js	1448536787
1.780	test_660156.js	1448639419
2.127	test_getAll.js	1448639419
3.188	test_getAll.js	1448637767
2.272	test_660156.js	1448643888
2.247	test_660156.js	1448545374
1.730	test_660156.js	1448499043
2.045	test_getAll.js	
1.571	test_660156.js	



Even better, every column
can be queried like an
edge in a data cube; each
row is a point in the cube

ActiveData

Slice the cube along the dimensions for humane presentation



ActiveData

Contents

- Unit Test Results – 5 billion records:
 - 75 dimensions, or more
 - Sub-minute response time
 - faster the second time (cached)
 - faster when scaled
- Perf – 50 million
- Buildbot job steps – 170 million

ActiveData

Built for analysis

- Public service endpoint
- Clean, Cartesian, data from JSON document store (read DataFrames).
- Columns annotated with type, measurement units, domain cardinality, and domain members*
- Distinction between dimensions and measures**

*still in prototype stage

**as defined by query

A Small Problem

A Small Problem

Lists, tables and grids are hard to read

ActiveData Query Tool


```
1 {
2   "from": "unittest",
3   "select": ["run.timestamp", "result.duration"],
4   "where": {"and": [
5     {"regex": {"result.test": ".*test_getAll.js"}},
6     {"gt": {"run.timestamp": "{{today-2day}}"}}
7   ]},
8   "limit": 10000
9 }
10
11
12
13
14
15
```

EXECUTE

```
{
  "from": "unittest",
  "select": ["run.timestamp", "result.duration"],
  "where": {"and": [
    {"regex": {"result.test": ".*test_getAll.js"}},
    {"gt": {"run.timestamp": "{{today-2day}}"}}
  ]},
  "limit": 10000
}
```

1872 rows (up to 3000 shown)

rownum	run.timestamp	result.duration
0	27-Nov-2015 16:58:09	36.74499988555908
1	27-Nov-2015 16:58:09	24.750999927520752
2	27-Nov-2015 17:05:12	11.331000089645386
3	27-Nov-2015 15:26:45	6.547999858856201
4	27-Nov-2015 15:26:45	2.99399995803833
5	27-Nov-2015 15:22:47	3.187999963760376



Solution

Automated Charting

Automated Charting

Information required is Accessible

- Access to columns' domain: categorical or algebraic
- Distinction between dimensions and measures
- Enough information to decide how best to show a query result
- Ranges that ignore those outliers

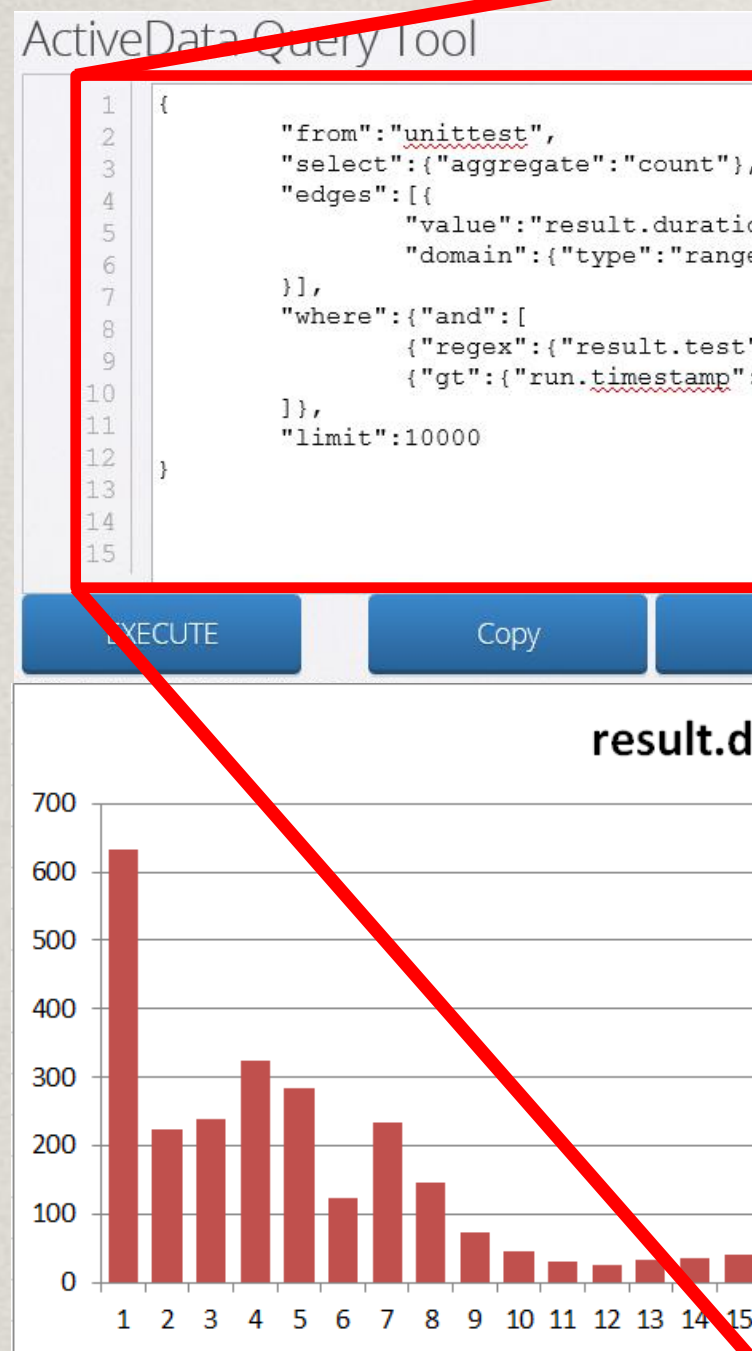
Automated Charting

Example



Automated Charting

Example



```
{
  "from": "unittest",
  "select": { "aggregate": "count" },
  "edges": [{
    "value": "result.duration",
    "domain": {
      "type": "range",
      "min": 0,
      "max": 100,
      "interval": 1
    }
  }],
  "where": { "and": [
    { "regex": { "result.test": ".*test_getAll.js" } },
    { "gt": { "run.timestamp": "{{today-2day}}" } }
  ] },
  "limit": 10000
}
```


The Big Problem

The Big Problem

Test results are consistent,
given time-of-day:

1. Possible slowdown caused by changeset
2. Retrigger multiple times, before and after, to confirm
3. All tests run at same time, giving consistent, but different, result.
4. Conclusion: Slowdown is real
5. Reality: Changeset code was never run!

[Bug 1190877](#)

The Big Problem

Test results are consistent,
given day-of-week:

1. Test infrastructure sees increase load on Monday
2. Performance tests slow down (???)
3. False positive alerts every Monday
4. Monday alerts are ignored
5. Blind to regressions introduced on Monday

The Big Problem

Good tests get disabled

- t-Test is can detect change in modal behaviour, if lucky.
- When unlucky, test is determined to “misbehave”, or have “excessive outliers”.
- “Misbehaving” tests get ignored or disabled

Attachments show [Bug 1196419](#) is detecting perf changes, yet there is suggestion to disable it.

The Big Problem

Not enough engineers

- Perf tests are generating 10's of alerts per-day, many more if thresholds were not set so high.
- Chasing a real regression requires backfill, and human time to track through the trees
- False alerts are consuming valuable time
- We are swamped ...
- We only just started ...

The Big Problem

Only just started?!

- Perf has 450 tests (in 54 suites)
- ActiveData has 30,000 (in 30 suites)
- Not just test times (which are boring)
 - Test Failure Rates
 - Machine Failure rates
 - Platform/BuildOptions
- Plus other dimensions from datasets we have not considered yet.

Non Solutions

- Fixing the tests
 - Too many
 - Nothing to fix – multithreaded, jit optimizations, gc, are unavoidable
 - Not in scope (for Metrics)
- Disabling tests
- Manually ensure the t-test is applicable
 - Too many
 - Character changes too fast

The Solution

Automated Change Detection!

Automated Change Detection!

Math Strategies?

- Mixture models for multimodal data?
- 2-state Markov model for weekends?
- Periodic model?
- Include estimates of models' accuracy
- When not enough data, more can be requested

Automated Change Detection!

Implementation?

- Unreasonably effective t-test – we are missing way to confirm the data matches its assumptions.
- Iterative Solution! – add one model at a time; ensuring we have a way to pick the best.
- Machine Learning? – too computationally intensive, given we already know how to solve.

Automated Change Detection!

Engineering?

- ActiveData manages ETL issues
- ActiveData can provide the data frames to analysis tools.
- [MozCI](#) to (re)trigger tests.
- Must handle backfilling data and the different conclusions that result (lifecycle).
- Feed conclusions to apps that UI humans (PerfHerder, AlertManager, dzAlerts)

The Future!

With Automated Change Detection

- Track regressions and improvements down to the changeset and bug.
- ...or no evidence, or not enough machine resources to make a conclusion
- Each Firefox release can list the bugs and sum regressions/improvements
- Sheriffs are not needed to manage regressions
- Find anomalies in slices we never considered!

End