simple performance regression pass/fail script

Document created by Ben England on Aug 8, 2016 • Last modified by Ben England on Apr 17, 2020
Version 5

Click on this link for the new version of the performance regression pass-fail script. This script which is based on python's scipy.stats.ttest_ind () function for applying a T-test to a set of sample, based on a method described in Raj Jain's classic "the Art of Computer Systems Performance Analysis". The script knows nothing about the system being tested, and assumes that the configuration and workload are the same, except for the software version. When you factor out comments and user interface, it's 10-20 lines of python. There are other implementations of this test - KVM engineering team used this method as well, as documented here.

The inputs are a one-sample-per-line format with a baseline sample set (for example, RHS 2.0) , and a "current" sample set (for example, RHS 2.0+). You have to specify at least 3 samples, preferably more. You then specify a sample type (throughput or response time), a confidence threshold for the T-test, and a maximum percentage deviation for the baseline and current sample sets.

If the T-test indicates that the probability is above your confidence threshold that the sample means are different, and if the current mean is "worse" (lower if throughput, higher if response time) than the baseline mean, then you can conclude that there was a performance regression. The script returns a process exit status code of 0 if the test passed and different non-zero codes for anything else.

To run the above script, you must have the **python3-scipy** and **python3-numpy** packages installed.

Here is an example of how you would use it -- the output of echo $? is showing you the exit status returned by the script; this exit code can be used by test automation scripts without looking at the output from the script. In this example we have a clear regression:

```
# echo base: `cat base.samples`  current: `cat test.samples`
base: 3.1 2.9 2.8 2.7 3.0 current: 2.0 2.1 2.5 2.3
# python3 is-regression-v2.py throughput 95 10 10 base.samples test.samples
decision parameters:
  sample type = throughput
  confidence threshold =  95.00 %
  max. pct. deviation =  10.00 %
  regression threshold =  10.00 %
sample stats for baseline:
  min = 2.700000
  max = 3.100000
  mean = 2.900000
  sd = 0.158114
  pct.dev. =  5.45 %
sample stats for current:
  min = 2.000000
  max = 2.500000
  mean = 2.225000
  sd = 0.221736
  pct.dev. =  9.97 %
current mean improvement over baseline is -23.28 percent
magnitude of change is at least 13.31%
t-test t-statistic = 5.351296 probability = 0.001063
t-test says that mean of two sample sets differs with probability  99.89%
probability that sample sets have same mean = 0.0011
declaring a performance regression test FAILURE because of lower throughput
# echo $?
10
```

And an example of a non-failure, the samples are lower, but not enough to be sure that a regression did occur.

```
# echo base: `cat base.samples`  current: `cat t2.samples`
base: 3.1 2.9 2.8 2.7 3.0 current: 2.7 2.65 2.5 2.85
# VERBOSE=1 python3 is-regression-v2.py throughput 95 10 0 base.samples t2.samples
decision parameters:
  sample type = throughput
  confidence threshold =  95.00 %
  max. pct. deviation =  10.00 %
  regression threshold =   0.00 %
5 samples read from file base.samples
[3.1, 2.9, 2.8, 2.7, 3.0]
sample stats for baseline:
  min = 2.700000
  max = 3.100000
  mean = 2.900000
  sd = 0.158114
  pct.dev. =  5.45 %
4 samples read from file t2.samples
[2.7, 2.65, 2.5, 2.85]
sample stats for current:
  min = 2.500000
  max = 2.850000
  mean = 2.675000
  sd = 0.144338
  pct.dev. =  5.40 %
current mean improvement over baseline is -7.76 percent
magnitude of change is at least  2.31%
t-test t-statistic = 2.201398 probability = 0.063600
```

t-test says that mean of two sample sets differs with probability 93.64%

probability that sample sets have same mean = 0.0636

probability threshold = 0.0500

sample sets are statistically indistinguishable for specified confidence level

echo $?

0

ATTACHMENTS

OUTCOMES

Helpful (1)

Visibility: Performance & Scale Community of Practice • 196 Views

Last modified on Apr 17, 2020 9:19 AM

Tags: rhcs_product_performance_scale | virtualization_product_performance_scale | networking_product_perfor | openshift_product_performance_scale | perf-cop | management_product_performance_scale | knowledge_base_product_performance_scale | emerging_product_performance_scale | database_product_ | perf_scale _product_performance_scale Edit tags

Categories: OpenShift | Virtualization | RHCS/OCS | Database | Emerging | Knowledge Base | Management | Networking | Perf &Scale Lab Edit categories

Global Reach

0%

Impact 41

Sentiment **Neutral** 3

3 Comments | 0 Author comments

Charles Shi
Dec 12, 2017 5:25 AM

The link "https://github.com/redhat-performance/perf-dept/blob/master/scripts/is-regression.py" in the beginning of this article seems to be unavailable now.

Actions Like • 0 Reply

Ben England
@ Charles Shi on Dec 12, 2017 7:57 AM

sorry this is in a private github, I forgot about that, it really should be in a public one. Will try to move it, but in the meantime I'll e-mail it to you and let me know if you find it useful or have comments on it.

Actions Like • 0 Reply

Charles Shi
@ Ben England on Dec 12, 2017 8:02 PM

Thank you! I got your email this morning. :-) I think I will use it in the near future. It should be helpful very much!

Actions Like • 0 Reply

Recommended Content

Roadmaps

Grace hopper contact

GraceHopper2020Banner

GraceHopperResources

Engineering All-Hands - September 10, 2020

Incoming Links

loss of edit privs for document