AI FOR TESTING DATA PROCESSES

PART 2

PyData Edinburgh 2018 • Lightning Talk • 4th April 2019

www.tdda.info/pdf/tdda-artie2-2019.pdf

Nicholas J. Radcliffe
Stochastic Solutions Limited

TESTING
DATA
PROCESSES

TESTING
DATA

CONSTRAINT
GENERATION
& VERIFICATION

TESTING
DATA PROCESSING

REFERENCE
TESTS

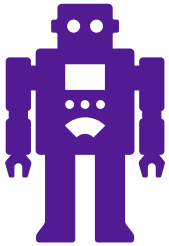
REFERENCE TESTS

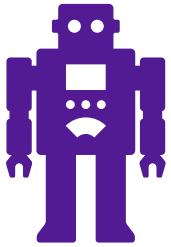
INPUTS → ANALYTICAL PROCESS → OUTPUTS

DATASETS, NUMBERS, GRAPHS, MODELS, DECISIONS ETC.

Record inputs 

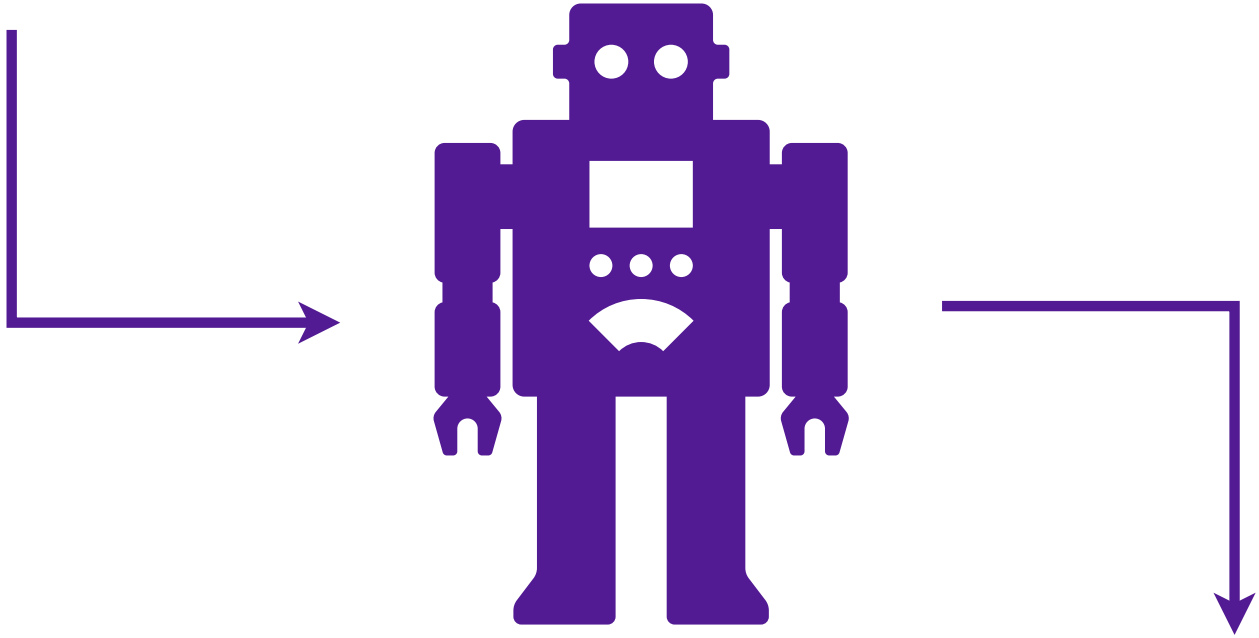
Capture as script 

Record ("reference") outputs 

Develop a verification procedure (`diff`) *and periodically rerun: do the same inputs (still) produce ~~the same~~ equivalent outputs?* 

ARTIE

`sh classify.sh`



test script **`test_sh_classify_sh.py`**

reference outputs **`ref/sh_classify_sh`**

example1.sh

```
echo "Hello, Edinburgh PyData!"
echo
echo "This is gentest, running on `hostname`"
echo
echo "I have to say, the weather was better in Münich!"
echo
echo "Today, `date` it's proper dreich here."
echo
echo "Let's have a file as well." > FILE1
echo
echo "Have a number: $RANDOM" >> FILE1
```

example1.sh

```
echo "Hello, Edinburgh PyData!"
echo
echo "This is gentest, running on `hostname`"
echo
echo "I have to say, the weather was better in Münich!"
echo
echo "Today, `date` it's proper dreich here."
echo
echo "Let's have a file as well." > FILE1
echo
echo "Have a number: $RANDOM" >> FILE1
```

TDDA Wizard

```
$ tdda gentest
Enter shell command to be tested: sh example1.sh
Enter name for test script [test_sh_example1_sh]:
Check all files written under $(pwd)?: [y]:
Enter other files to be checked, one per line, then blank line:

Check stdout?: [y]:
Check stderr?: [y]:
Exit code should be zero?: [y]:
Number of times to run script?: [2]:
```

Wizard Output

```
Running command 'sh example1.sh' to generate output (run 1 of 2).
Saved (non-empty) output to stdout to /Users/njr/tmp/pydata/ref/sh_example1_sh/STDOUT.
Saved (empty) output to stderr to /Users/njr/tmp/pydata/ref/sh_example1_sh/STDERR.
Copied $(pwd)/FILE1 to $(pwd)/ref/sh_example1_sh/FILE1
Running command 'sh example1.sh' to generate output (run 2 of 2).
Saved (non-empty) output to stdout to /Users/njr/tmp/pydata/ref/sh_example1_sh/2/STDOUT.
Saved (empty) output to stderr to /Users/njr/tmp/pydata/ref/sh_example1_sh/2/STDERR.
Copied $(pwd)/FILE1 to $(pwd)/ref/sh_example1_sh/2/FILE1
Test script written as /Users/njr/tmp/pydata/test_sh_example1_sh.py


Command execution took: 0.027s


SUMMARY:

Directory to run in:   /Users/njr/tmp/pydata
Shell command:         sh example1.sh
Test script generated: test_sh_example1_sh
Reference files:
    $(pwd)/FILE1
Check stdout:          yes (was 9 lines)
Check stderr:          yes (was empty)
Expected exit code:    0
```

Generated Code

```
$ cat /Users/njr/tmp/pydata/test_sh_example1_sh.py
# -*- coding: utf-8 -*-

"""
test_sh_example1_sh.py: Automatically generated test code
from tdda gentest.

Generation command:

  tdda gentest 'sh example1.sh' 'test_sh_example1_sh.py' '.'
STDOUT STDERR
"""

from __future__ import absolute_import
from __future__ import print_function
from __future__ import division

import os
import sys

from tdda.referencetest import ReferenceTestCase
from tdda.referencetest.gentest import exec_command

COMMAND = 'sh example1.sh'
CWD = os.path.abspath(os.path.dirname(__file__))
REFDIR = os.path.join(CWD, 'ref', 'sh_example1_sh')
```

Note exclusions for local context and run-to-run variability

```
class TestAnalysis(ReferenceTestCase):
    @classmethod
    def setUpClass(cls):
        (cls.output,
         cls.error,
         cls.exc,
         cls.exit_code,
         cls.duration) = exec_command(COMMAND, CWD)

    def test_no_exception(self):
        msg = 'No exception should be generated'
        self.assertEqual((str(self.exc), msg), ('None', msg))

    def test_exit_code(self):
        self.assertEqual(self.exit_code, 0)

    def test_stdout(self):
        substrings = [
            'godel.local',
            '9 Apr 2019 17:45:49',
        ]
        self.assertStringCorrect(self.output,
                                 os.path.join(REFDIR, 'STDOUT'),
                                 ignore_substrings=substrings)

    def test_stderr(self):
        self.assertStringCorrect(self.error,
                                 os.path.join(REFDIR, 'STDERR'))

    def test_FILE1(self):
        patterns = [
            r'^Have a number\: \d{4,5}$',
        ]
        self.assertFileCorrect(os.path.join(CWD, 'FILE1'),
                               os.path.join(REFDIR, 'FILE1'),
                               ignore_patterns=patterns)

if __name__ == '__main__':
    ReferenceTestCase.main()
```

Saved Files

```
$ ls ref/sh_example1_sh/
2   FILE1  STDERR STDOUT

$ more ref/sh_example1_sh/FILE1
Let's have a file as well.
Have a number: 9310

$ more ref/sh_example1_sh/STDOUT
Hello, Edinburgh PyData!

This is gentest, running on godel.local

I have to say, the weather was better in Münich!

Today, Tue  9 Apr 2019 17:45:49 BST it's proper dreich
here.

$ more ref/sh_example1_sh/STDERR
$
```

(This file has no content)

Running the Tests

```
$ python test_sh_example1_sh.py
.....
----------------------------------------------------------------------
Ran 5 tests in 0.018s

OK
```

Running Repeatedly

If you run enough times, you will get a failure, because the exclusion is assuming the random number generated will always be four or five digits.

On the night, it didn't fail.

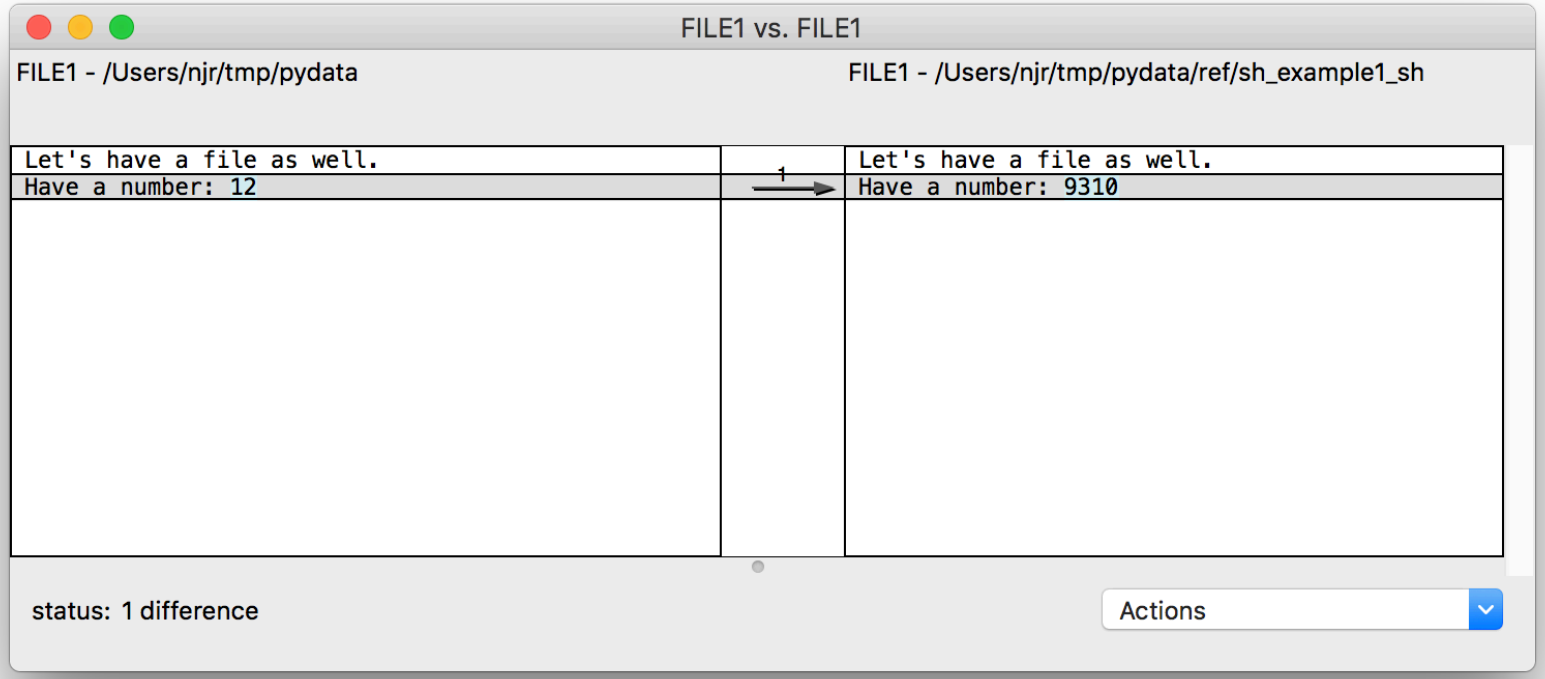
But after, I ran it another 33 times, and the last time it failed.

When it does fail

```
$ python test_sh_example1_sh.py
1 line is different, starting at line 2
Compare with:
    diff /Users/njr/tmp/pydata/FILE1 /Users/njr/tmp/pydata/ref/sh_example1_sh/FILE1

Note exclusions:
    ignore_patterns:
        ^Have a number\: \d{4,5}$
F....
======================================================================
FAIL: test_FILE1 (__main__.TestAnalysis)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "test_sh_example1_sh.py", line 61, in test_FILE1
    ignore_patterns=patterns)
  File "/Users/njr/python/tdda/tdda/referencetest/referencetest.py", line 857, in assertTextFileCorrect
    self._check_failures(failures, msgs)
  File "/Users/njr/python/tdda/tdda/referencetest/referencetest.py", line 1046, in _check_failures
    self.assert_fn(failures == 0, msgs.message())
AssertionError: 1 line is different, starting at line 2
Compare with:
    diff /Users/njr/tmp/pydata/FILE1 /Users/njr/tmp/pydata/ref/sh_example1_sh/FILE1

Note exclusions:
    ignore_patterns:
        ^Have a number\: \d{4,5}$

----------------------------------------------------------------------
Ran 5 tests in 0.018s

FAILED (failures=1)
```

And if you run the diff:

```
$ opendiff /Users/njr/tmp/pydata/FILE1 /Users/njr/tmp/pydata/ref/sh_example1_sh/FILE1
```



It is indeed that `\d{4,5}` *is too specific to capture all the variation.*
(In this case, it's just two digits!)
Easily fixed by hand.

njr@StochasticSolutions.com

http://tdda.info

https://github.com/tdda

#tdda*

*tweet (DM) us email address for invitation
Or email me.

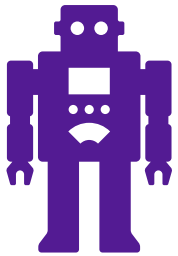
@tdda0 @njr0 @StochasticSolns

Correct interpretation: Zero

Error of interpretation: Letter "Oh"

pip install tdda

(gentest coming soon!)



www.tdda.info/pdf/tdda-artie2-2019.pdf