Nosetests → Pytest

Motivation

Writing Tests

Running Tests

Motivation

Writing Tests

Running Tests

Pytest is Actively Maintained

"… [Nose] will likely cease without a new person/team to take over maintainership."

– Nose Maintainers

nose.readthedocs.io/en/latest/index.html#note-to-users

Pytest Found Deprecation Warnings

```
- np.random.random_integers(a, b)
+ np.random.randint(a, b + 1)

- with self.assertRaises(Exception) as c:
-     ...
-     self.assertEqual(c.exception.message, …)
+ with self.assertRaisesRegexp(Exception, "^msg$"):
+     ...
```

Pytest Asserts Are More Pythonic

`unittest`	`pytest`
`self.assertEqual(a, b)`	`assert a == b`

Motivation

Writing Tests

Running Tests

Assertions

unittest	pytest
`self.assertEqual(a, b)`	`assert a == b`
`self.assertLessThan(a, b)`	`assert a < b`
`self.assertIn(a, my_list)`	`assert a in my_list`
`self.assertRaises(Exception):`	`pytest.raises(Exception):`

No Constructors in Tests

```
class TestTmp(object):

    def __init__(self):
        # setup logic

    def test_0(self):
        x = func_under_test()
        assert x == 2
```

No Constructors in Tests

```
class TestTmp(object):

    def __init__(self):
        # setup logic

    def test_0(self):
        x = func_under_test()
        assert x == 2
```

Setup and Teardown

UnitTest Style

```
class TestA(TestCase):

    def setUp(self):
        ...
    def tearDown(self):
        ...
    def setUpClass(self):
        ...
    def tearDownClass(self):
        ...
```

Pytest / Nose Style

```
class TestA:

    def setup_method(self):
        ...
    def teardown_method(self):
        ...
    def setup_class(self):
        ...
    def teardown_class(self):
        ...
```

Pytest Fixtures

```
@pytest.fixture(scope="class")  # created once per class
def load_data():
    # setup code
    yield '{"foo": "bar"}'  # if no teardown, use return
    # teardown code

def test_analyze(load_data):
    analysis_results = analyze(load_data())
    assert results == ["myResults"]
```

Test Discovery

- Pytest loads based on:
  - Modules: `test_*.py` or `*_test.py`
  - Classes: `Test*`
  - Functions (outside class or in test class): `test_*`

Test Discovery

- Pytest loads based on:
  - Modules: `test_*.py` or `*_test.py`
  - Classes: `Test*`
  - Functions (outside class or in test class): `test_*`
- Example: Naming integration tests file
  - `test_workflow.py`
  - `workflow_test.py`
  - `integration_test_workflow.py`

Motivation

Writing Tests

Running Tests

Running Tests

```
$ pytest
```

Test Output: Passing

```
$ pytest
====================================== test session starts ======================================
platform darwin -- Python 2.7.16, pytest-4.6.5, py-1.5.4, pluggy-0.13.0
benchmark: 3.2.2 (defaults: timer=time.time disable_gc=False min_rounds=5 min_time=0.000005
max_time=1.0 calibration_precision=10 warmup=False warmup_iterations=100000)
rootdir: /path/to/repository/root/, inifile: pytest.ini
plugins: benchmark-3.2.2, cov-2.8.1
collected 111 items

models/ecoli/tests/test_arrow.py .                                                        [  0%]
path/to/second/test.py .....                                                              [ 10%]
...
=============================== 109 passed, 2 skipped in 7.29 seconds ===========================
```

Failures

```
class TestTmp(object):

    def test_0(self):
        x = 1
        assert x == 2

    def test_1(self):
        x = 1
        assert 0 < x < 1

    def test_2(self):
        x = {1: 2}
        assert 5 in x
```

```
    def test_0(self):
        x = 1
>       assert x == 2
E       assert 1 == 2
test_tmp.py:5: AssertionError

    def test_1(self):
        x = 1
>       assert 0 < x < 1
E       assert 1 < 1
test_tmp.py:9: AssertionError

    def test_2(self):
        x = {1: 2}
>       assert 5 in x
E       assert 5 in {1: 2}
test_tmp.py:13: AssertionError
```

Key Takeaways

- Writing Tests
  - Use `assert`
  - Use `setup_method`, not `__init__`
- Running Tests
  - Update dependencies
  - Run `pytest`, not `nosetests` (change in PyCharm)

More Information

pytest.org

PR#728