binomial_distributions

January 21, 2019

1 Binomial Distributions

Use NumPy to create simulations and compute proportions for the following outcomes. The first one is done for you.

In [1]:
```
# import numpy
import numpy as np
```

1.0.1 1. A fair coin flip produces heads

In [2]:
```
# simulate 1 million tests of one fair coin flip
# remember, the output of these tests are the # successes, or # heads
tests = np.random.binomial(1, 0.5, int(1e6))

# proportion of tests that produced heads
(tests == 1).mean()
```

Out[2]:
```
0.49954100000000001
```

1.0.2 2. Five fair coin flips produce exactly one head

In [3]:
```
# simulate 1 million tests of five fair coin flips
tests = np.random.binomial(5, 0.5, int(1e6))

# proportion of tests that produced 1 head
(tests == 1).mean()
```

Out[3]:
```
0.15586700000000001
```

1.0.3 3. Ten fair coin flips produce exactly four heads

In [4]:
```
# simulate 1 million tests of ten fair coin flips
tests = np.random.binomial(10, 0.5, int(1e6))

# proportion of tests that produced 4 heads
(tests == 4).mean()
```

Out[4]:
```
0.20591999999999999
```

1.0.4 4. Five biased coin flips with P(H) = 0.8 produce exactly five heads

In [5]:
```
# simulate 1 million tests of five biased coin flips
tests = np.random.binomial(10, 0.8, int(1e6))

# proportion of tests that produced 5 heads
(tests == 5).mean()
```

Out[5]: 0.026294000000000001

1.0.5 5. Ten biased coin flips with P(H) = 0.15 produce at least 3 heads

In [6]:
```
# simulate 1 million tests of ten biased coin flips
tests = np.random.binomial(10, 0.15, int(1e6))

# proportion of tests that produced at least 3 heads
(tests == 3).mean()
```

Out[6]: 0.129858

In [ ]: