MonteCarloExample

February 16, 2021

We know that the area of a circle with radius r is πx^2. So the fraction of a square of side length s that is contained in an inscribed circle is $\pi/4$.

Estimate the probability a random point in $[0, 1] \times [0, 1]$ is in the unit circle. Multiply by 4.

[3]:
```python
import numpy as np

n = 100000000

result = np.square(np.random.rand(n, 2)).sum(axis = 1) < 1
print("pi is approximately", np.average(result)*4)
print("Error is approximately ", np.sqrt(np.var(result)/n)*16)
```

```
pi is approximately 3.14183496
Error is approximately  0.0006568059590621017
```

Assume that when a baseball player is at bat, their chances of hitting are i.i.d. Assume 300 at bats per season. What is the chance that a player with a 25% of hitting the ball hits strictly more balls than a player with a 30% chance of hitting the ball

[2]:
```python
n = 1000000
results = np.random.binomial(300, .25, n) >= np.random.binomial(300, .30, n)
print("Probability is approximately ", np.average(results))
print("Error is approximately ", np.sqrt(np.var(results)/n))
```

```
Probability is approximately  0.091729
Error is approximately  0.0002886430157807391
```

Let's assume a league has 100 player in it. Only 1 of the players is a .300 player, the remaining are .250 players. If a player hits above a .300, are is the chance they are the .300 player? Hint, use Bayes Theorem.

[4]:
```python
n = 1000000

Prob_Bad_Looks_Good = np.average(np.random.binomial(300, .25, n) >= 90)

Prob_Good_Looks_Good = np.average(np.random.binomial(300, .3, n) >= 90)

Prob_Good_given_Looks_Good = (Prob_Good_Looks_Good * .01) /␣
 ↪(Prob_Good_Looks_Good * .01 + Prob_Bad_Looks_Good * .99)
```

```
print("Probability a random player with average over .300 is the good player is␣
 ↪", Prob_Good_given_Looks_Good)
```

```
Probability a random player with average over .300 is the good player is
0.1579083242065437
```

A line has an infinite number of people in it; all have i.i.d birthdays distributed uniformly thoughout a 365 day year. Starting at the beginning of the line, each person declares the month and day or their birthday. What is the expected number of people that need to declare their birthdays before 3 people share the same birthday?

```
[6]: n = 100000

results = np.zeros(n)

for i in range(n):
    days = np.zeros(365)
    j = 0
    while(i):
        j += 1
        days[np.random.randint(365)] += 1
        if max(days) >= 3:
            results[i] = j
            break

print("Average is approximately", np.average(results))
print("Error is approximately", np.sqrt(np.var(results)/n))
```

```
Average is approximately 88.63731
Error is approximately 0.10361317030010711
```