

```
!pip install -U future
```

```
Requirement already satisfied: future in /usr/local/lib/python3.10/dist-packages (0.18.3)
Collecting future
  Downloading future-1.0.0-py3-none-any.whl (491 kB)
    _____ 491.3/491.3 kB 3.5 MB/s eta 0:00:00
Installing collected packages: future
  Attempting uninstall: future
    Found existing installation: future 0.18.3
    Uninstalling future-0.18.3:
      Successfully uninstalled future-0.18.3
  Successfully installed future-1.0.0
```

```
from __future__ import print_function, division
from builtins import range
```

```
import numpy as np
import matplotlib.pyplot as plt
```

```
class Bandit:
    def __init__(self, m):
        self.m = m
        self.mean = 0
        self.N = 0
    def pull(self):
        return np.random.randn() + self.m
    def update(self, x):
        self.N += 1
        self.mean = (1 - 1.0/self.N)*self.mean + 1.0/self.N*x
```

```
def run_experiment (m1, m2, m3, eps, N):
    bandits = [Bandit(m1), Bandit(m2), Bandit (m3)]
    data = np.empty (N)
    for i in range(N):
        # epsilon greedy
        p = np.random.random()
        if p < eps:
            j = np.random.choice(3)
        else:
            j = np.argmax([b.mean for b in bandits])
        x = bandits[j].pull()
        bandits[j].update(x)

        # for the plot
        data[i] = x
    cumulative_average = np.cumsum(data)/ (np.arange(N) + 1)
    #plot moving average ctr
    plt.plot(cumulative_average)
    plt.plot(np.ones(N)*m1)
    plt.plot(np.ones(N)*m2)
    plt.plot(np.ones(N)*m3)
    plt.xscale('log')
    plt.show()
    for b in bandits:
        print(b.mean)
    return cumulative_average
```

```
if __name__ == '__main__':
    c_1 = run_experiment (1.0, 2.0, 3.0, 0.1, 100000)
    c_05 = run_experiment(1.0, 2.0, 3.0, 0.05, 100000)
    c_01 = run_experiment (1.0, 2.0, 3.0, 0.01, 100000)
    #log scale plot
    plt.plot(c_1, label='eps = 0')
    plt.plot(c_05, label='eps = 0.05')
    plt.plot(c_01, label='eps = 0.1')
    plt.plot(c_05, label='eps = 0.3')
    plt.plot(c_05, label='eps = 0.5')
    plt.legend()
    plt.xscale('log')
    plt.show()
    # linear plot
    plt.plot(c_1, label='eps = 0')
    plt.plot(c_05, label='eps = 0.05')
    plt.plot(c_01, label='eps = 0.1')
    plt.plot(c_05, label='eps = 0.3')
    plt.plot(c_05, label='eps = 0.5')
    plt.legend()
    plt.show()
```



