Upper Confidence Bound

May 24, 2024

1 Core to Multi-armed Bandit Problem: Optimality When Facing Uncertainty

$$\mathrm{UCB}_i(t) = \hat{\mu}_i(t) + \sqrt{\frac{2\log t}{n_i(t)}}$$

$$R(T) = \mathbb{E}\left[\sum_{i=0}^{T} X^* - \sum_{i=0}^{T} X_i\right] = T \cdot \mu^* - \sum_{t=1}^{T} X_t$$

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
```

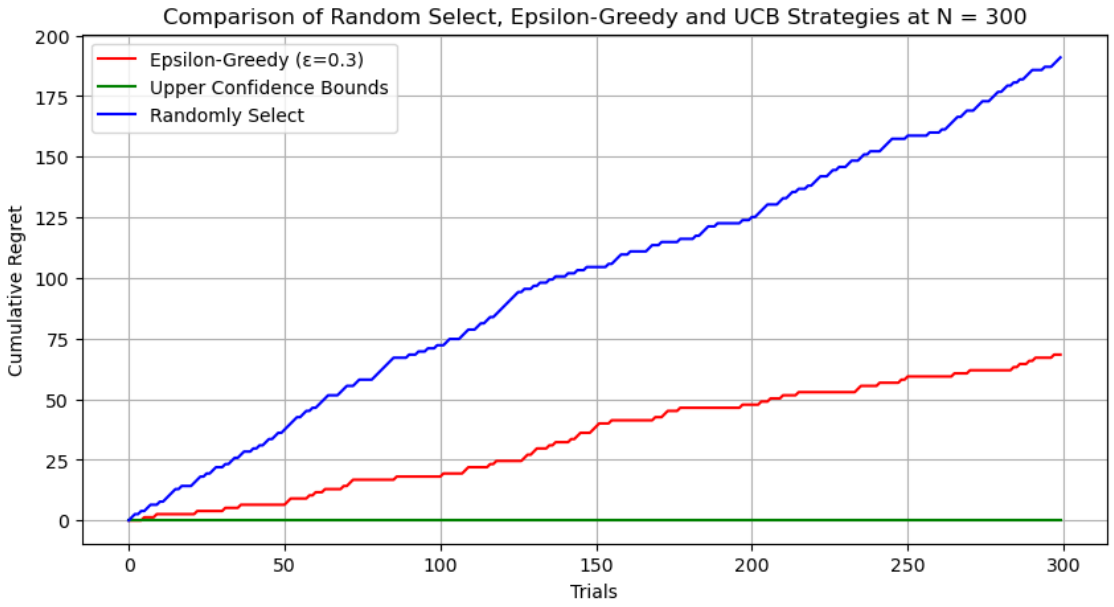
[ ]:
```
n_trials = 300
probabilities = np.random.normal(size=2)
n_arms = len(probabilities)

counts = np.zeros(n_arms)
values = np.zeros(n_arms)
total_regret_ucb = np.zeros(n_trials)
total_regret_eps = np.zeros(n_trials)
total_regret_rand = np.zeros(n_trials)

def random_select():
    return np.random.randint(n_arms)

def epsilon_greedy(epsilon):
    if np.random.rand() < epsilon:
        return np.random.randint(n_arms)
    return np.argmax(values)

def ucb(t):
    if np.min(counts) == 0:
        return np.argmin(counts)
    confidence_bounds = np.sqrt(2 * np.log(t) / counts)
    return np.argmax(values + confidence_bounds)
```

1.0.1 Run the simulation for both strategies

[ ]:
```
# Compare to optimal option for calculating regret
best_prob = np.max(probabilities)

for t in range(1, n_trials + 1):
    # Random Selection
    arm_rand = random_select()
    reward_rand = np.random.rand() < probabilities[arm_rand]
    regret_rand = best_prob - probabilities[arm_rand]
    counts[arm_rand] += 1
    values[arm_rand] += (reward_rand - values[arm_rand]) / counts[arm_rand]
    total_regret_rand[t-1] = total_regret_rand[t-2] + regret_rand if t > 1 else␣
 ↪regret_rand

    # Epsilon-Greedy Strategy
    epsilon = 0.3
    arm_eps = epsilon_greedy(epsilon)
    reward_eps = np.random.rand() < probabilities[arm_eps]
    regret_eps = best_prob - probabilities[arm_eps]
    counts[arm_eps] += 1
    values[arm_eps] += (reward_eps - values[arm_eps]) / counts[arm_eps]
    total_regret_eps[t-1] = total_regret_eps[t-2] + regret_eps if t > 1 else␣
 ↪regret_eps

    # UCB Strategy
    if t == 1:
        counts.fill(0)
        values.fill(0)
    arm_ucb = ucb(t)
    reward_ucb = np.random.rand() < probabilities[arm_ucb]
    regret_ucb = best_prob - probabilities[arm_ucb]
    counts[arm_ucb] += 1
    values[arm_ucb] += (reward_ucb - values[arm_ucb]) / counts[arm_ucb]
    total_regret_ucb[t-1] = total_regret_ucb[t-2] + regret_ucb if t > 1 else␣
 ↪regret_ucb
```

[ ]:
```
plt.figure(figsize=(10, 5))
plt.plot(total_regret_eps, label='Epsilon-Greedy ( =0.3)', color='red')
plt.plot(total_regret_ucb, label='Upper Confidence Bounds', color='green')
plt.plot(total_regret_rand, label='Randomly Select', color='blue')
plt.xlabel('Trials')
plt.ylabel('Cumulative Regret')
plt.title(f'Comparison of Random Select, Epsilon-Greedy and UCB Strategies at N␣
 ↪= {n_trials}')
plt.legend()
plt.grid(True)
```

```
plt.show()
```



[ ]:
```
n_trials = 2000
probabilities = np.random.normal(size=2)
n_arms = len(probabilities)
counts = np.zeros(n_arms)
values = np.zeros(n_arms)
total_regret_ucb = np.zeros(n_trials)
total_regret_eps = np.zeros(n_trials)
total_regret_rand = np.zeros(n_trials)

# Compare to optimal option for calculating regret
best_prob = np.max(probabilities)

for t in range(1, n_trials + 1):
    # Epsilon-Greedy Strategy
    epsilon = 0.3
    arm_eps = epsilon_greedy(epsilon)
    reward_eps = np.random.rand() < probabilities[arm_eps]
    regret_eps = best_prob - probabilities[arm_eps]
    counts[arm_eps] += 1
    values[arm_eps] += (reward_eps - values[arm_eps]) / counts[arm_eps]
    total_regret_eps[t-1] = total_regret_eps[t-2] + regret_eps if t > 1 else␣
 ↪regret_eps

    # UCB Strategy
```

```
    if t == 1:
        counts.fill(0)
        values.fill(0)
    arm_ucb = ucb(t)
    reward_ucb = np.random.rand() < probabilities[arm_ucb]
    regret_ucb = best_prob - probabilities[arm_ucb]
    counts[arm_ucb] += 1
    values[arm_ucb] += (reward_ucb - values[arm_ucb]) / counts[arm_ucb]
    total_regret_ucb[t-1] = total_regret_ucb[t-2] + regret_ucb if t > 1 else␣
↪regret_ucb
```

[ ]:
```
plt.figure(figsize=(10, 5))
plt.plot(total_regret_eps, label='Epsilon-Greedy ( =0.3)', color='red')
plt.plot(total_regret_ucb, label='Upper Confidence Bounds', color='green')
plt.xlabel('Trials')
plt.ylabel('Cumulative Regret')
plt.title(f'Comparison of Epsilon-Greedy and UCB Strategies at N={n_trials}')
plt.legend()
plt.grid(True)
plt.show()
```