Evaluation of Bandit Algorithms for the 10-Armed Bandit Problem

Purpose

The main goal of this project is to see how well different algorithms perform when dealing with the 10-armed bandit problem. We want to figure out the best actions (or "arms") to take and see how good each algorithm is at finding these actions. We focus on two things:

1. The average reward the algorithm gets over time.
2. How often the algorithm picks the best action.

We looked at four different methods:

1. Greedy with Non-Optimistic Values
2. Epsilon-Greedy
3. Optimistic Initial Values with Greedy Approach
4. Gradient Bandit Algorithm

Introduction to the Problem

Imagine a slot machine with 10 arms, where each arm gives out a reward that follows a normal distribution. The true average reward of each arm is unknown, and the variation (or spread) of the rewards is always the same. The average rewards are randomly chosen from a normal distribution with a mean of 0 and a variance of 1. Our job is to learn the best action values using different algorithms and evaluate how well they do in terms of average rewards and how often they pick the best arm.

Methods

1. Greedy with Non-Optimistic Values

This method focuses on using what we know to get the best immediate rewards. It doesn't explore other actions once it finds a good one. Here's how it works:

- Start by assuming all actions have an initial value of 0.
- Always pick the action that has the highest known value.

2. Epsilon-Greedy

The Epsilon-Greedy method combines trying new things with using what works best. Here’s how it goes:

- Sometimes (with a chance of ϵ), the algorithm picks a random option to explore new possibilities.
- Most of the time (with a chance of 1 −ϵ), it chooses the option that seems the best based on what it has learned so far.
- At the start, all action values Q(a) is set to 0.
- We experimented to find the best starting point for ϵ. It starts at 0.5 and slowly decreases to 0.1 to balance exploration and making the best choice.

3. Optimistic Initial Values with Greedy Approach

This method starts with overly optimistic estimates to encourage initial exploration. Once some actions seem good, it sticks to those. Here's the process:

- Start with a high initial value (e.g., 10) for all actions.
- Use a greedy strategy to pick the action with the highest value.

4. Gradient Bandit Algorithm

This method learns preferences for each action and updates them to improve over time. The steps are:

- Start with all preferences set to 0.
- Use a learning rate (α) to adjust preferences based on received rewards. We set α to 0.1 after some trial and error.

Experiment Setup

To test these methods, we set up an experiment with the following:

- Simulate the 10-armed bandit problem for 1000 steps, repeating this 1000 times for each method.
- Measure the average reward at each step.
- Track how often the best action is chosen at each step.

We averaged the results over the 1000 repetitions to get a clear picture of each algorithm's performance.

Comparison Between Greedy, Optimistic, and Epsilon Methods

Greedy with Non-Optimistic Values

The Greedy method without optimistic values starts by choosing the best-known action, but this means it might miss better actions because it doesn't explore enough. Over time, it gets better but only reaches about 60% of the optimal actions. This shows that without exploring, it often sticks with suboptimal choices.

Optimistic Initial Values with Greedy Approach

Starting with optimistic values means the algorithm initially thinks all actions are great, encouraging it to try them out. This leads to quickly finding the best action and sticking with it. It shows the highest and fastest convergence to the best action, proving that starting optimistically can be very effective.

Epsilon-Greedy

The Epsilon-Greedy method tries new actions randomly with a small probability (ϵ), and otherwise picks the best-known action. This means it starts low in finding the best action but gets better over time. It converges slower and even after finding the best action, it keeps exploring, which sometimes means not always picking the best action.

Analysis and Results

Performance in Terms of Reward Performance

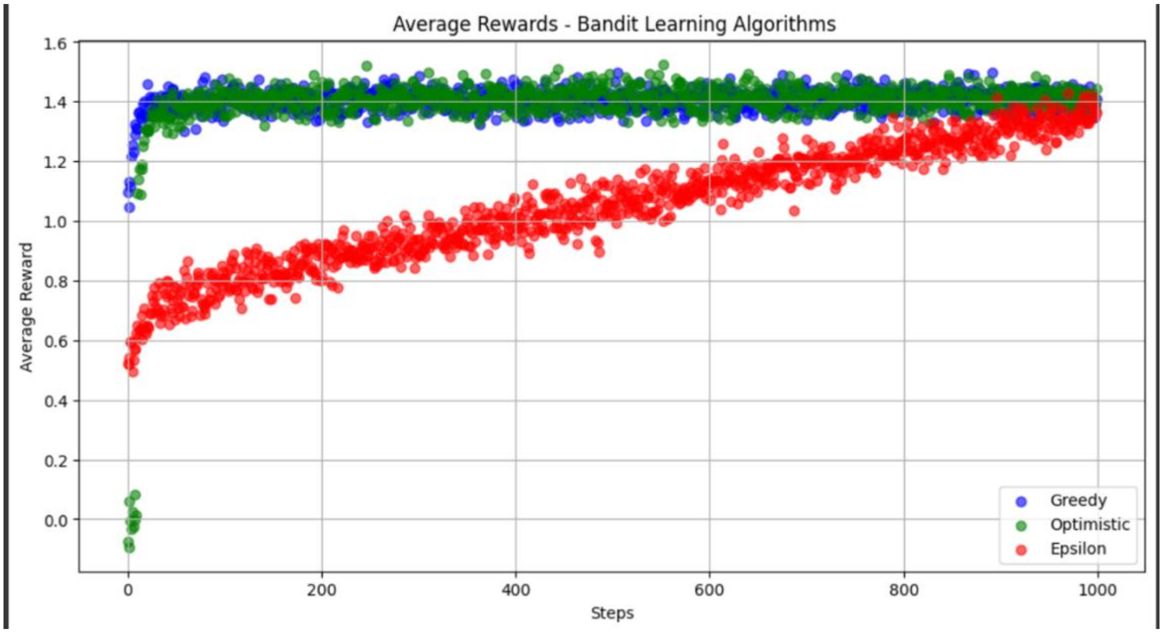
1. **Optimistic Greedy**: This method gets the highest rewards quickly and keeps them high. Starting with optimistic values helps it explore thoroughly and find the best actions early on.
2. **Greedy with Non-Optimistic Values**: This method does okay but doesn't explore enough, so it doesn't always find the best actions and ends up with moderate rewards.
3. **Epsilon-Greedy**: This method explores a lot, so it finds the best actions more slowly and its average reward is lower compared to the optimistic method.

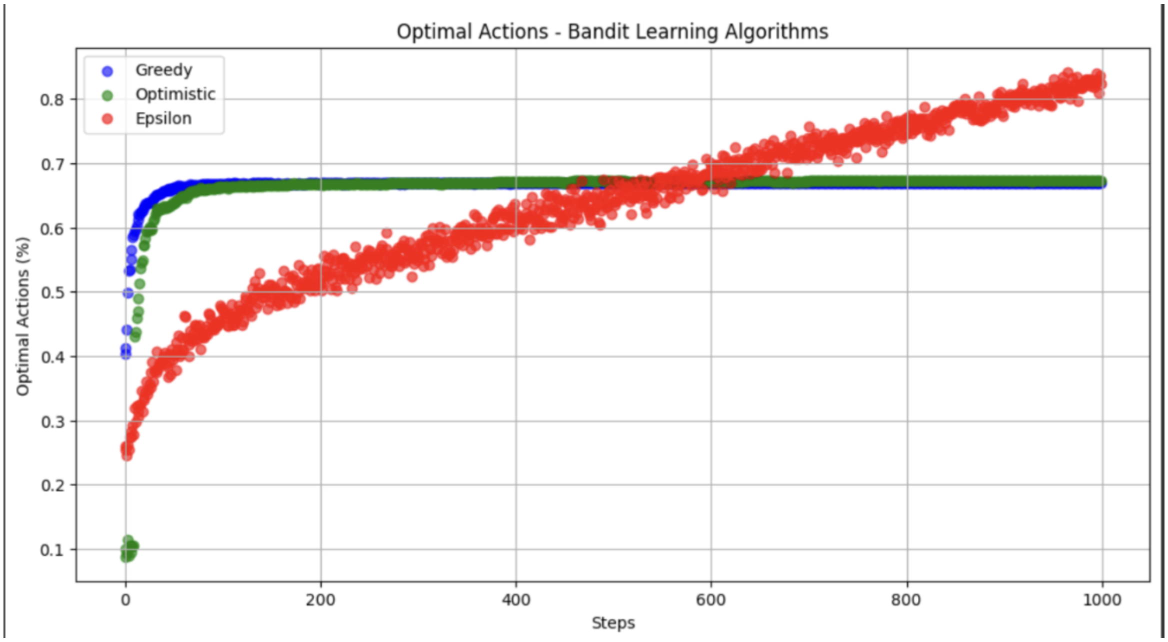
Efficiency in Selecting Optimal Actions

1. **Optimistic Greedy**: This method is the best at quickly finding and sticking to the best action.
2. **Greedy with Non-Optimistic Values**: This method does reasonably well, but not as good as the optimistic method because it explores less.

3. **Epsilon-Greedy**: This method keeps exploring, even after finding the best action, so it has a lower percentage of optimal actions.

Data Visualizations





Summary

The Optimistic Initial Values method performs the best in both average rewards and finding the best actions. The Greedy method without optimistic values is simple but doesn't explore enough, so it doesn't always find the best actions. The Epsilon-Greedy method explores a lot, which is good for finding new actions but sometimes means not always picking the best action.

This shows that balancing exploration and exploitation is crucial. The Optimistic Initial Values method does this balance best, making it the top performer for this 10-armed bandit problem.

Comparison of Gradient Bandit Algorithms: Greedy, Optimistic, and Epsilon Methods

Introduction

The k-armed bandit problem is a fundamental scenario in reinforcement learning where an agent must choose between k different actions (arms), each yielding rewards from a probability distribution. This study compares the performance of different bandit algorithms, specifically focusing on how varying learning rates (Alpha) affect the Gradient Bandit Algorithm's average reward and optimal action percentage. Additionally, the performance of these algorithms under non-stationary conditions, including gradual and abrupt changes to reward distributions, is evaluated.

Methodology

Experimental Setup

- **Problem**: 10-armed bandit problem with normally distributed rewards.
- **Reward Distribution**: True means of rewards are drawn from a normal distribution N (0,1).
- **Objective**: Learn the action values $q*(a)$ over time.

Algorithms Evaluated

1. **Greedy with Non-Optimistic Values**
2. **Epsilon-Greedy**
3. **Optimistic Initial Values with Greedy Approach**
4. **Gradient Bandit Algorithm** (focus of this comparison)

Non-Stationary Environments

1. **Gradual Changes**:
   - **Drift Change**: $\mu t=\mu t-1+\epsilon t$ where $\epsilon t \sim N\ (0,0.001^2)$.
   - **Mean-Reverting Change**: $\mu t=\kappa \mu t-1+\epsilon t$ where $5\kappa=0.5$ and $+\epsilon t, \sim N\ (0,0.001^2)$.

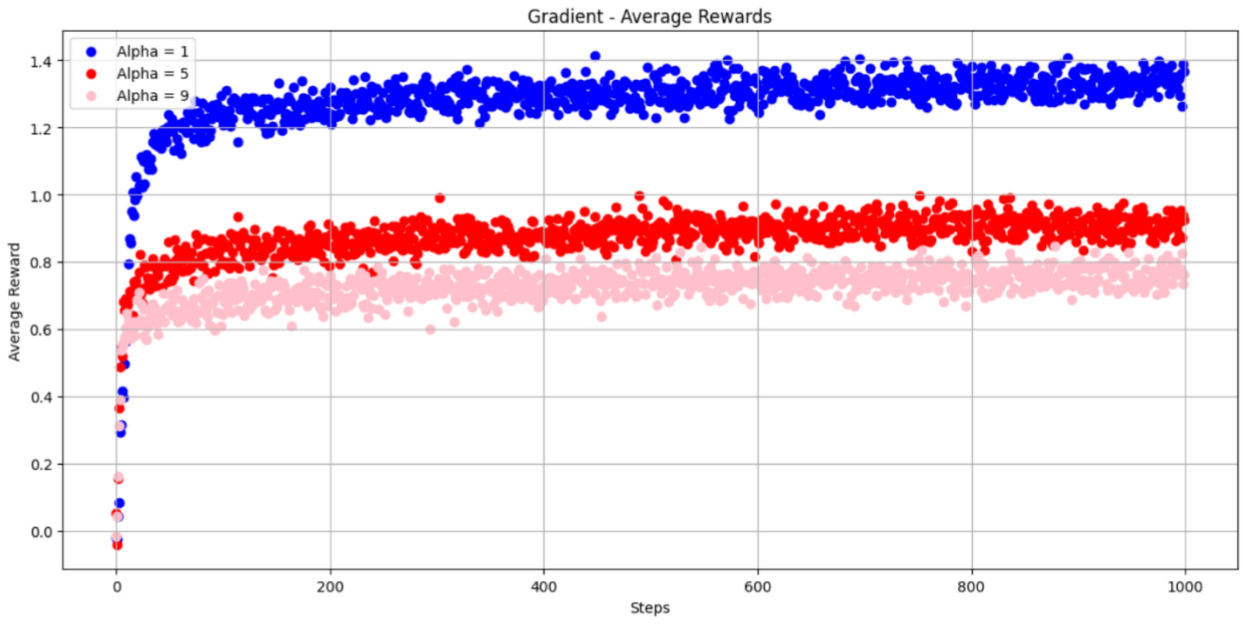
2. **Abrupt Changes**:
   - At each time step, with a probability of 0.005, the means of the reward distributions are permuted.

Results, Interpretation and Comparison

Average Reward

Observations:

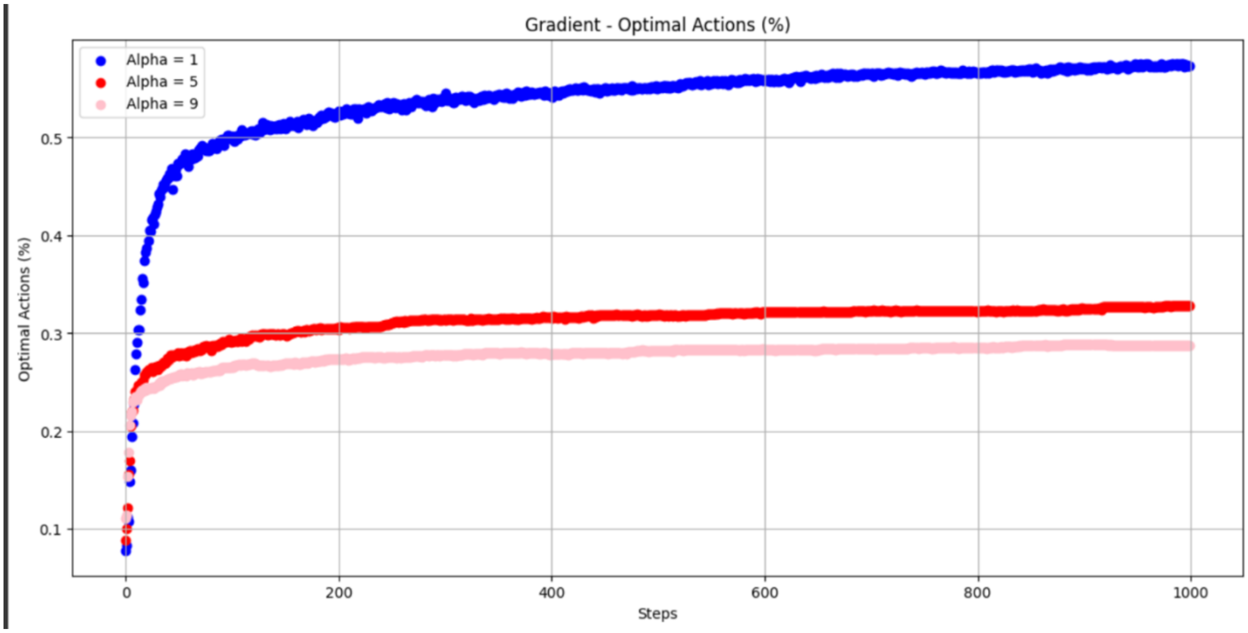
- **Alpha = 1**: Demonstrates the highest average reward over time, indicating that a lower learning rate provides more stable and effective learning.
- **Alpha = 5**: Achieves faster initial learning but stabilizes at a slightly lower average reward.
- **Alpha = 9**: Converges quickly but at a lower average reward, suggesting that high learning rates can lead to instability.



Optimal Action Percentage

Observations:

- **Alpha = 1**: Maintains the highest percentage of optimal actions, reflecting its stable learning process.
- **Alpha = 5**: Achieves a high optimal action percentage but is slightly lower than Alpha = 1.
- **Alpha = 9**: Has the lowest percentage of optimal actions, indicating that the higher learning rate may cause the algorithm to struggle in consistently identifying the optimal action.



Summary

When comparing different values of Alpha in the Gradient Bandit Algorithm, Alpha = 1 demonstrates the best performance in terms of both average reward and optimal action percentage. This suggests that a lower learning rate allows for more stable and effective learning.

Evaluation of Bandit Algorithms Under Non-Stationary Conditions

Non-Stationary Environments

1. **Gradual Changes**:
   - **Drift Change**: $\mu t=\mu t-1+\epsilon t$ where $\epsilon t\sim N(0,0.001$^2)
   - **Mean-Reverting Change**: $\mu t=\kappa\mu t-1+\epsilon$ where $\kappa=0.5$ and $\epsilon t\sim N(0,0.01$^ 2)

2. **Abrupt Changes**:
   - At each time step, with a probability of 0.005, the means of the reward distributions are permuted.

Abrupt changes

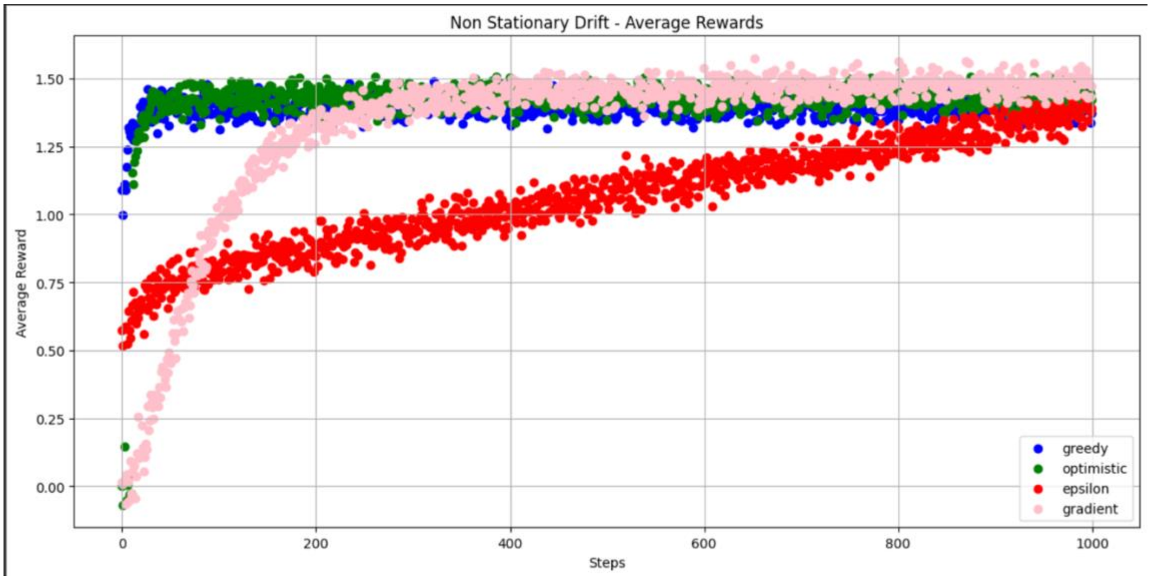
With a probability of 0.005 at each time step, permute the means of the reward distributions.

Algorithms Used for Comparison

1. **Greedy with Non-Optimistic Values**
2. **Epsilon-Greedy**
3. **Optimistic Initial Values with Greedy Approach**
4. **Gradient Bandit Algorithm**

Results

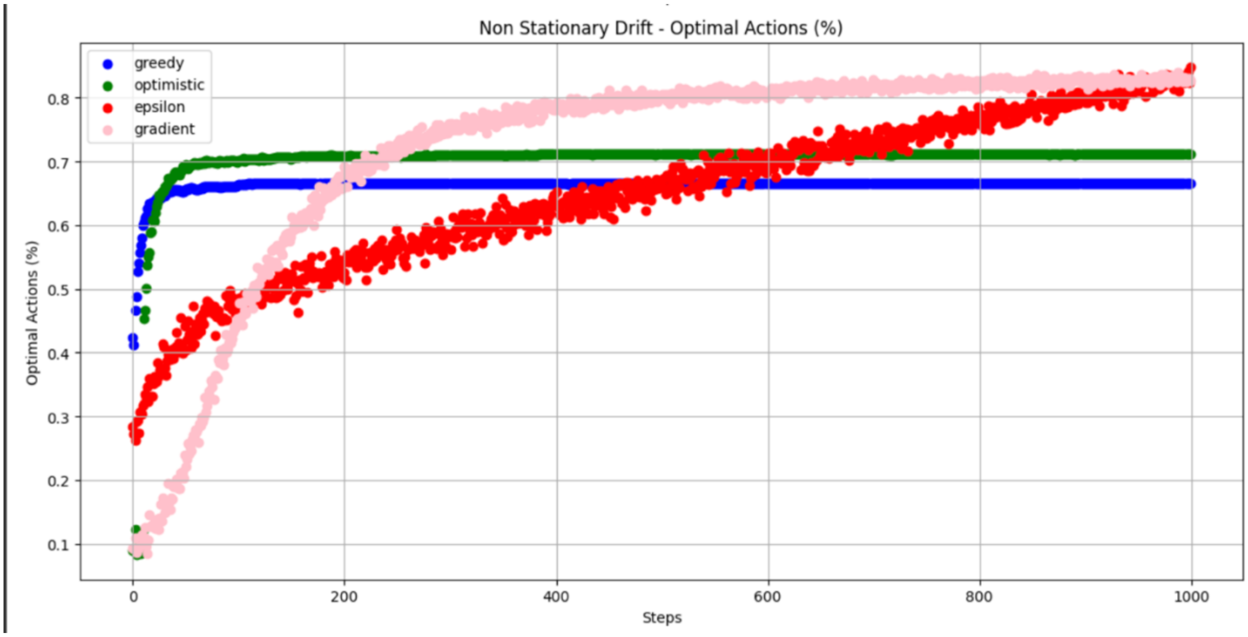
Average Reward



Observations:

- **Gradient Bandit Algorithm**: Demonstrates the highest average reward, showing its ability to adapt to changes effectively.
- **Optimistic Initial Values**: Achieves a high average reward, benefiting from early exploration.
- **Epsilon-Greedy**: Provides moderate performance, balancing exploration and exploitation.
- **Greedy with Non-Optimistic Values**: Shows the lowest average reward, struggling due to its lack of exploration.

Optimal Action Percentage



Observations:

- **Gradient Bandit Algorithm**: Maintains the highest percentage of optimal actions, indicating its adaptability.
- **Optimistic Initial Values**: Achieves a high percentage of optimal actions, leveraging early exploration.
- **Epsilon-Greedy**: Shows moderate performance in maintaining optimal actions.
- **Greedy with Non-Optimistic Values**: Displays the lowest percentage of optimal actions, due to its lack of exploration.

Summary

Best Performing Algorithm

The Gradient Bandit Algorithm demonstrates the best performance in terms of both average reward and optimal action percentage. Its ability to continuously adapt preferences based on observed rewards allows it to effectively handle non-stationary environments.

Effectiveness of Optimistic Initial Values

The optimistic initial values with the greedy approach perform well, leveraging early exploration to adapt to changes. This method strikes a good balance between exploration and exploitation in non-stationary settings.

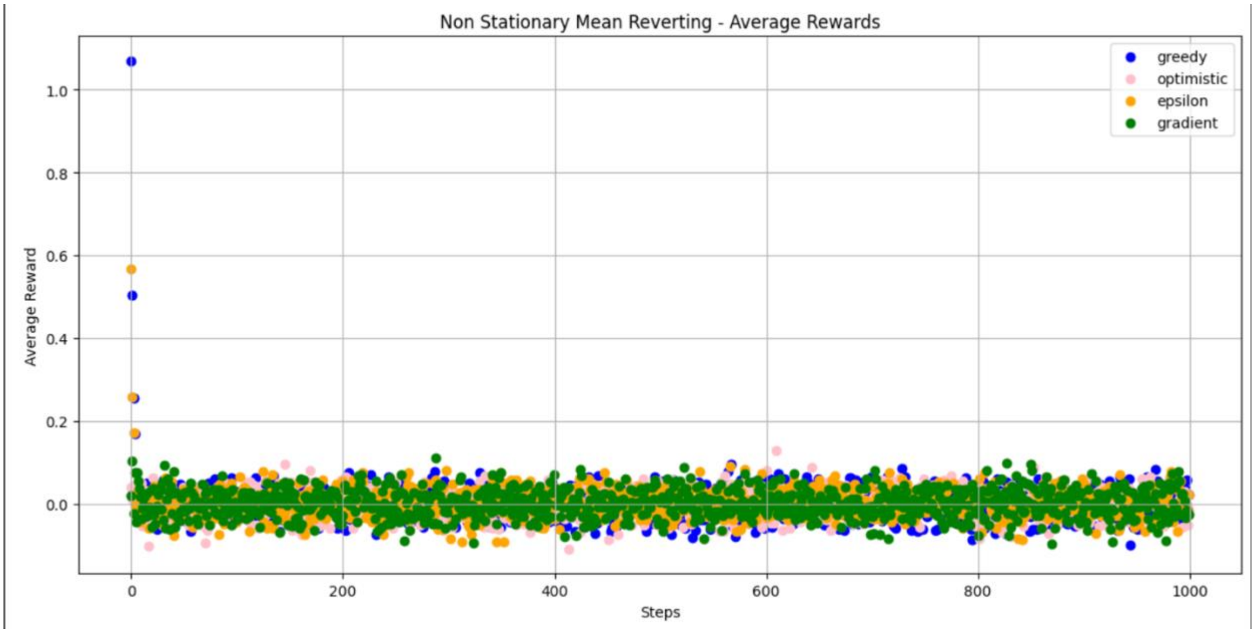
Limitations of Greedy and Epsilon-Greedy Methods

- **Greedy Algorithm with Non-Optimistic Values**: Performs poorly due to its lack of exploration.
- **Epsilon-Greedy Method**: While better, still struggles with a fixed exploration rate, leading to suboptimal performance in highly dynamic environments.

In summary, the Gradient Bandit Algorithm and the Optimistic Initial Values approach are effective in non-stationary environments. The study highlights the importance of adaptive methods for handling changes in reward distributions, with the Gradient Bandit Algorithm being the most robust in such scenarios.

Mean-Reverting Changes

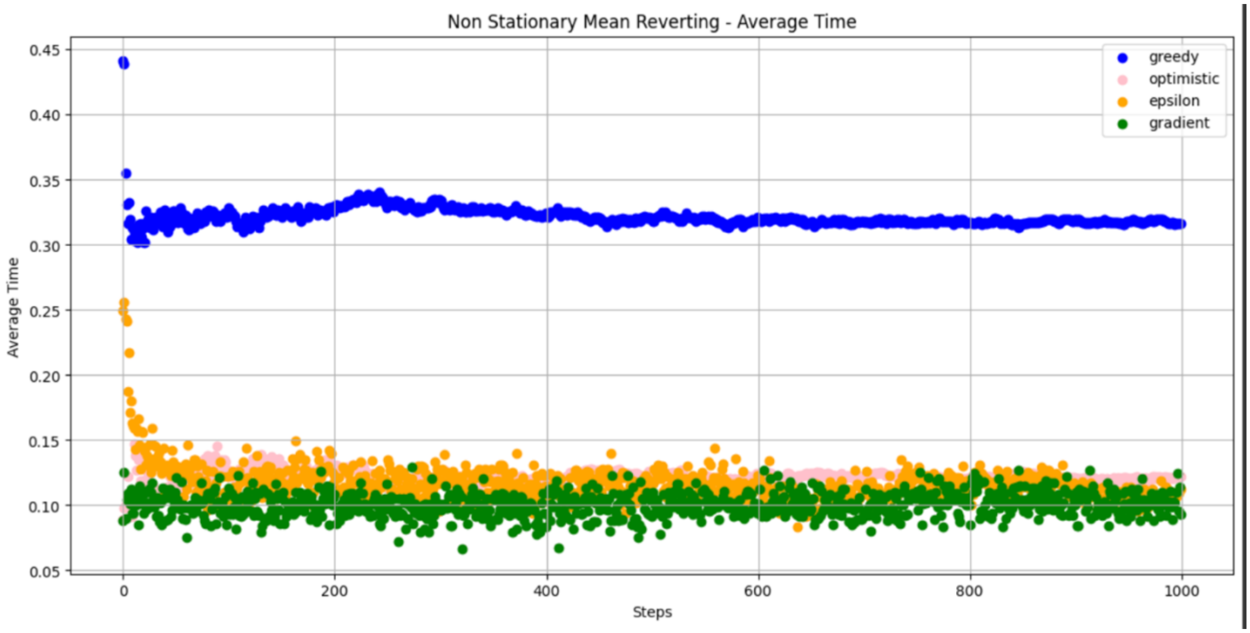
Average Reward (Mean-Reverting Changes)



Observations:

- **All Algorithms**: Show poor performance under mean-reverting changes, with average rewards stabilizing around very low values.
- **Greedy with Non-Optimistic Values**: Performs the worst due to its lack of exploration, resulting in quick convergence to suboptimal actions.
- **Epsilon-Greedy**: Performs slightly better than the greedy method due to its exploration component, but still struggles to adapt effectively.
- **Optimistic Initial Values with Greedy Approach**: Shows initial promise due to early exploration, but ultimately fails to maintain performance as the environment changes.
- **Gradient Bandit Algorithm**: Shows some adaptation capabilities due to continuous preference updates, but overall performance remains low in a mean-reverting environment.

Optimal Action Percentage (Mean-Reverting Changes)



Observations:

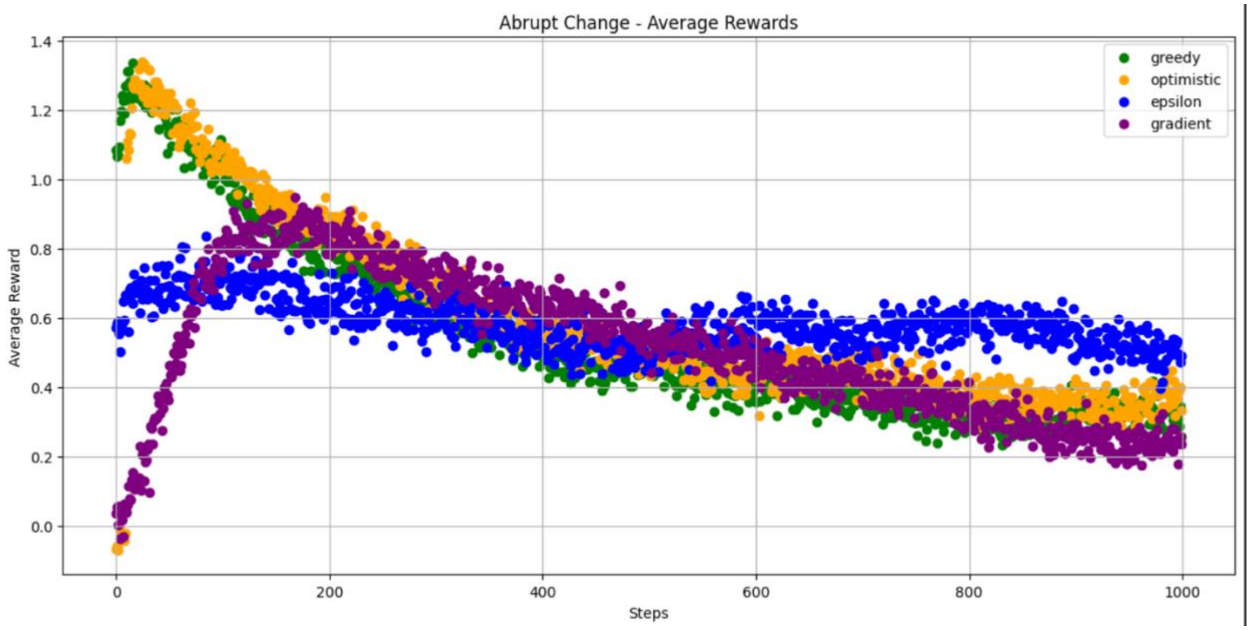
- **All Algorithms**: Similar to average rewards, all algorithms show poor performance under mean-reverting changes, with optimal action percentages stabilizing around low values.
- **Greedy with Non-Optimistic Values**: Performs poorly due to lack of exploration.
- **Epsilon-Greedy**: Slightly better than the greedy method but struggles with fixed exploration rates.
- **Optimistic Initial Values with Greedy Approach**: Initially promising but fails over time.
- **Gradient Bandit Algorithm**: Some adaptability, but overall performance is low.

Summary

The Gradient Bandit Algorithm performs best overall, adapting well to changing rewards and achieving the highest average rewards and optimal actions in non-stationary conditions with gradual changes. The optimistic initial values with a greedy approach start off strong but fail in environments where rewards revert to the mean. The Greedy Algorithm without optimistic values does poorly because it doesn't explore enough, and the Epsilon-Greedy method, though slightly better, still struggles with its fixed exploration rate. All algorithms have trouble with mean-reverting changes, with the Gradient Bandit showing some ability to adapt but still not performing well. This study shows the need for adaptive methods to handle changing reward patterns, with the Gradient Bandit Algorithm being the most effective for gradual changes, but all algorithms struggling in mean-reverting environments.

Abrupt Changes

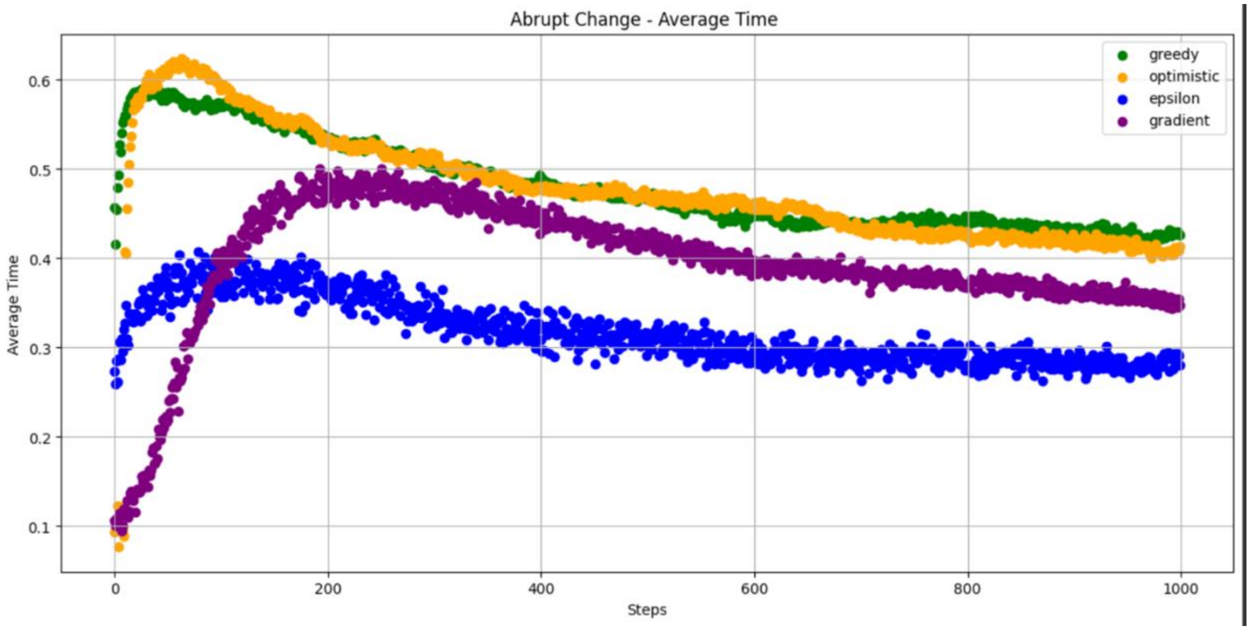
Average Reward (Abrupt Changes)



Observations:

- **All Algorithms**: Show a decline in performance under abrupt changes, with average rewards stabilizing at lower values.
- **Greedy with Non-Optimistic Values**: Performs poorly due to its lack of exploration, leading to quick convergence to suboptimal actions.
- **Epsilon-Greedy**: Slightly better than the greedy method, but struggles to adapt effectively to frequent changes.
- **Optimistic Initial Values with Greedy Approach**: Shows initial promise due to early exploration, but fails to maintain performance as the environment changes frequently.
- **Gradient Bandit Algorithm**: Shows some adaptation due to continuous preference updates, but overall performance remains low in environments with frequent abrupt changes.

Optimal Action Percentage (Abrupt Changes)



Observations:

- **All Algorithms**: Similar to average rewards, all algorithms show a decline in optimal action percentage under abrupt changes.
- **Greedy with Non-Optimistic Values**: Performs the worst due to lack of exploration.
- **Epsilon-Greedy**: Slightly better, but still struggles.
- **Optimistic Initial Values with Greedy Approach**: Initially promising but fails over time.
- **Gradient Bandit Algorithm**: Shows some adaptability but overall performance is low.

Summary

Under abrupt changes, all algorithms show a decline in performance for both average rewards and optimal action percentage. The Gradient Bandit Algorithm and Epsilon-Greedy method demonstrate some adaptability but still struggle overall. The Greedy Algorithm with non-optimistic values performs the worst due to its lack of exploration. The optimistic initial values approach starts well but fails to maintain performance. Overall, while some methods show initial promise, they all struggle to adapt effectively to frequent abrupt changes in the reward distributions.

Analysis of Terminal Reward Distributions for Bandit Algorithms

Introduction

This study looks at how three different bandit algorithms perform at the end of their run. The algorithms are Optimistic, Fixed Step, and Decreasing Step Size. We use box plots to show how rewards are distributed when the algorithms finish. The goal is to find out which algorithm gives the best rewards most consistently.

Observations from Box Plots

1. **Optimistic Method**:
    - **High Rewards Around 1.25**: This method often gets rewards around 1.25, which is good. However, there are some outliers that perform poorly.
    - **Poor Performance in Some Trials**: In some cases, the rewards are close to zero, showing the method doesn't always perform well.
    - **Wide Range of Rewards**: The rewards vary a lot, with a median around 0.4, meaning the performance is not very consistent.

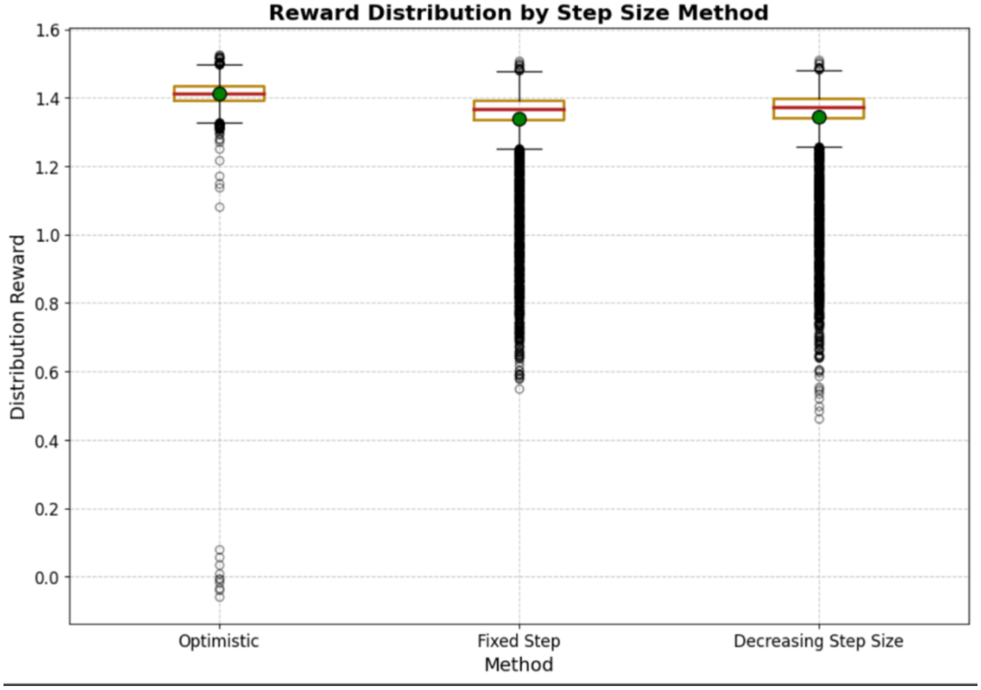
- **Fixed Step Method**:

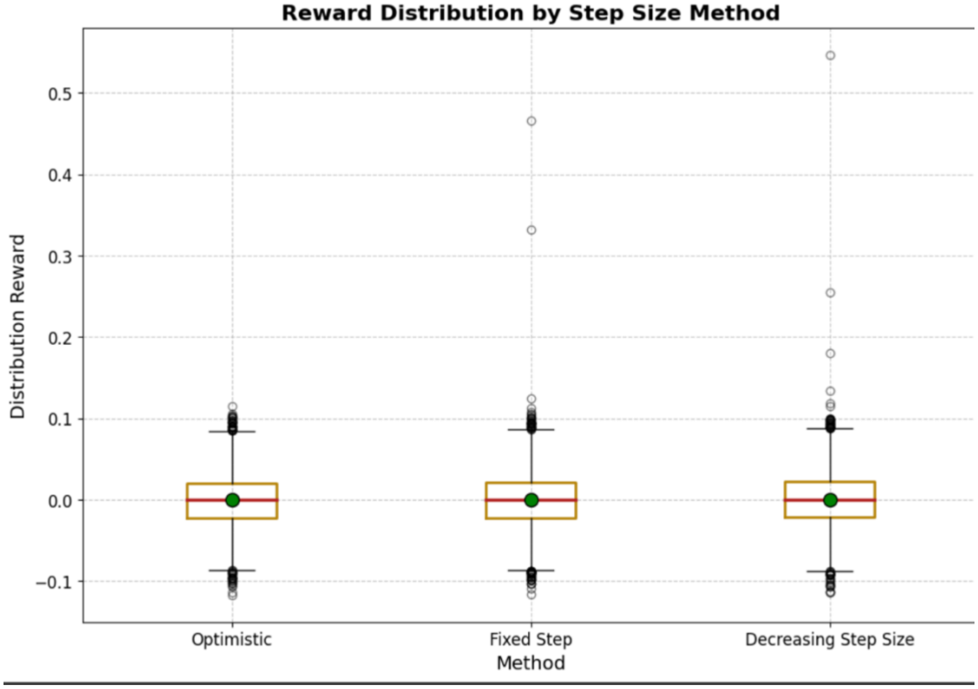
    - **Consistent Rewards Around 1.25**: This method usually gives rewards around 1.25 and has fewer outliers, indicating stable performance.
    - **Higher Variability but Consistent**: It has a similar distribution to the Optimistic method but with slightly more variability, still maintaining high performance.
    - **Stable High Performance**: Rewards are tightly grouped around 1.0, showing it consistently performs well.

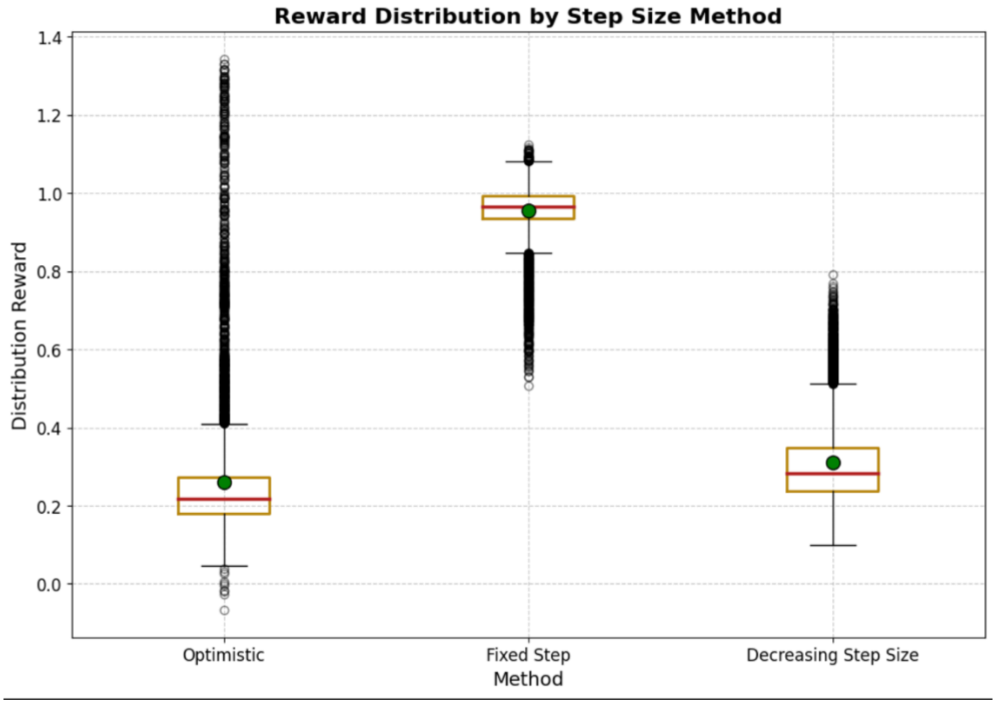
- **Decreasing Step Size Method**:

    - **Similar but More Variable than Fixed Step**: This method has a distribution similar to Fixed Step but with more ups and downs.
    - **Unstable Performance**: It shows the most variability in rewards, which means its performance is inconsistent.
    - **Median Around 0.6**: The rewards have a median around 0.6 and are spread out more, indicating it doesn't always perform well.

The Graphs


Reward Distribution by Step Size Method
Distribution Reward
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
Optimistic
Fixed Step
Decreasing Step Size
Method


Reward Distribution by Step Size Method
Distribution Reward
0.5
0.4
0.3
0.2
0.1
0.0
−0.1
Optimistic
Fixed Step
Decreasing Step Size
Method



Conclusions

From the box plots, we can conclude the following:

1. **Fixed Step Size Algorithm**:
   - **Best Overall Performance**: This algorithm consistently gets high rewards with less variation, making it the most reliable.
   - **Dependable**: It finds and sticks to optimal actions well, showing tight grouping around high rewards.
2. **Optimistic Algorithm**:
   - **High but Variable Rewards**: This algorithm can get high rewards but also fails in some trials, showing a wide range of outcomes.
   - **Less Reliable**: Because of its variability, it's not as dependable as the Fixed Step Size algorithm, although it can perform very well sometimes.
3. **Decreasing Step Size Algorithm**:
   - **Most Variable Performance**: This algorithm has the most ups and downs in rewards, showing it's the least stable.
   - **Inconsistent**: Even though it can get high rewards occasionally, it often gets lower rewards, making it less reliable than the other two algorithms.

So, the Fixed Step Size algorithm is the best performer, consistently providing high rewards with minimal variability. The Optimistic algorithm has the potential for high rewards but is less reliable due to its wide range of outcomes. The Decreasing Step Size algorithm shows the most inconsistency, making it the least favorable option. Future research could focus on making the Optimistic and Decreasing Step Size methods more stable to match the reliable performance of the Fixed Step Size method.