Capítulo 2: Multi-armed Bandits

Introdução

O problema do multi-armed bandit é uma versão simplificada do problema completo de aprendizado por reforço. É como o problema completo de aprendizado por reforço no sentido de que envolve aprender uma política, mas também é como nossa versão do problema do k-armed bandit no sentido de que cada ação afeta apenas a recompensa imediata. Se as ações forem permitidas a afetar a próxima situação bem como a recompensa, então temos o problema completo de aprendizado por reforço.

No problema do k-armed bandit, você está repetidamente enfrentado com uma escolha entre k ações diferentes. Após cada escolha, você recebe uma recompensa numérica escolhida de uma distribuição de probabilidade estacionária que depende da ação que você selecionou. Seu objetivo é maximizar a recompensa total esperada ao longo de algum período de tempo, por exemplo, ao longo de 1000 seleções de ação, ou passos de tempo.

Esta é a forma original do problema do k-armed bandit, assim nomeado por analogia a uma máquina caça-níqueis, ou "one-armed bandit", exceto que ele tem k alavancas em vez de uma. Cada seleção de ação é como jogar uma das alavancas da máquina caça-níqueis, e as recompensas são os pagamentos por acertar o jackpot. Através de seleções repetidas de ação, você deve maximizar seus ganhos concentrando suas ações nas melhores alavancas.

Exemplos do Mundo Real

O problema do multi-armed bandit aparece em muitas situações práticas:

1. **Testes A/B em websites**: Uma empresa quer testar diferentes versões de uma página web (k versões). Cada "braço" representa uma versão diferente, e a "recompensa" é a taxa de conversão ou cliques.
2. **Alocação de investimentos**: Um investidor deve decidir como alocar capital entre k diferentes ativos financeiros. Cada ativo tem um retorno esperado desconhecido, e o objetivo é maximizar o retorno total.
3. **Seleção de tratamentos médicos**: Um médico deve escolher entre k tratamentos diferentes para uma doença. A "recompensa" é a eficácia do tratamento, que varia entre pacientes.
4. **Recomendação de conteúdo**: Plataformas como Netflix ou YouTube devem escolher qual conteúdo recomendar aos usuários. Cada tipo de conteúdo é um "braço" e a recompensa é o engajamento do usuário.
5. **Otimização de anúncios online**: Sistemas de publicidade devem decidir qual anúncio mostrar a cada usuário, equilibrando anúncios que sabem que funcionam bem com novos anúncios que podem ter melhor desempenho.

Em nosso problema do k-armed bandit, cada uma das k ações tem uma recompensa esperada ou média dado que essa ação é selecionada; vamos chamar isso de valor dessa ação. Denotamos a ação selecionada no passo de tempo t como At, e a recompensa correspondente como Rt. O valor então de

uma ação arbitrária a, denotado q*(a), é a recompensa esperada dado que a é selecionada:

$$q^*(a) = E[R_t \mid A_t = a]$$

Se você conhecesse o valor de cada ação, então seria trivial resolver o problema do k-armed bandit: você sempre selecionaria a ação com maior valor. Assumimos que você não conhece os valores das ações com certeza, embora possa ter estimativas. Denotamos o valor estimado da ação a no passo de tempo t como Qt(a). Gostaríamos que Qt(a) fosse próximo de q*(a).

Se você mantém estimativas dos valores das ações, então a qualquer passo de tempo há pelo menos uma ação cujo valor estimado é maior. Chamamos essas de ações gananciosas (greedy actions). Quando você seleciona uma dessas ações, dizemos que você está fazendo **aproveitamento** (exploitation) do seu conhecimento atual dos valores das ações. Se em vez disso você seleciona uma das ações não-gananciosas, então dizemos que você está fazendo **exploração** (exploration), porque isso permite que você melhore sua estimativa do valor da ação não-gananciosa. Aproveitamento (exploitation) é a coisa certa a fazer para maximizar a recompensa esperada em um passo, mas exploração (exploration) pode produzir maior recompensa total a longo prazo.

Exemplo prático - Dilema exploração vs aproveitamento: Imagine um restaurante que você frequenta. Você sabe que o prato A é sempre bom (aproveitamento/exploitation), mas há um prato B novo no cardápio que pode ser ainda melhor. Pedir o prato B é exploração (exploration) - você pode descobrir algo melhor, mas também pode ter uma refeição pior. Este é o dilema fundamental dos multi-armed bandits.

2.2 Métodos de Valor de Ação

Começamos observando mais de perto métodos para estimar os valores das ações e para usar as estimativas para tomar decisões de seleção de ação, que coletivamente chamamos de métodos de valor de ação. Lembre-se de que o valor verdadeiro de uma ação é a recompensa média quando essa ação é selecionada. Uma maneira natural de estimar isso é calculando a média das recompensas realmente recebidas:

Qt(a) = (soma das recompensas quando a foi tomada antes de t) / (número de vezes que a foi tomada antes de t)

$$Q_t(a) = \left(\Sigma_{i=1 \text{ a } t-1} R_i \cdot 1\{A_i=a\}\right) / \left(\Sigma_{i=1 \text{ a } t-1} 1\{A_i=a\}\right)$$

onde 1{predicado} denota a variável aleatória que é 1 se o predicado for verdadeiro e 0 se não for. Se o denominador for zero, então definimos Qt(a) como algum valor padrão, como 0. À medida que o denominador vai para o infinito, pela lei dos grandes números, Qt(a) converge para q*(a). Chamamos isso de método de média amostral (sample-average method) para estimar valores de ação porque cada estimativa é uma média da amostra de recompensas relevantes.

A regra de seleção de ação mais simples é selecionar uma das ações com o maior valor estimado, isto é, uma das ações gananciosas (greedy actions) como

definidas na seção anterior. Escrevemos este método de seleção de ação gananciosa (greedy action selection) como:

At = argmax_a Qt(a)

onde argmax_a denota a ação a para a qual a expressão que segue é maximizada (com empates quebrados arbitrariamente). A seleção de ação gananciosa (greedy action selection) sempre faz aproveitamento (exploitation) do conhecimento atual para maximizar a recompensa imediata; não gasta tempo amostrando ações aparentemente inferiores para ver se elas podem realmente ser melhores. Uma alternativa simples é se comportar de forma gananciosa na maior parte do tempo, mas de vez em quando, digamos com pequena probabilidade ε, em vez disso selecionar aleatoriamente entre todas as ações com probabilidade igual, independentemente das estimativas de valor de ação. Chamamos métodos usando esta regra de seleção de ação quase-gananciosa de métodos ε-greedy.

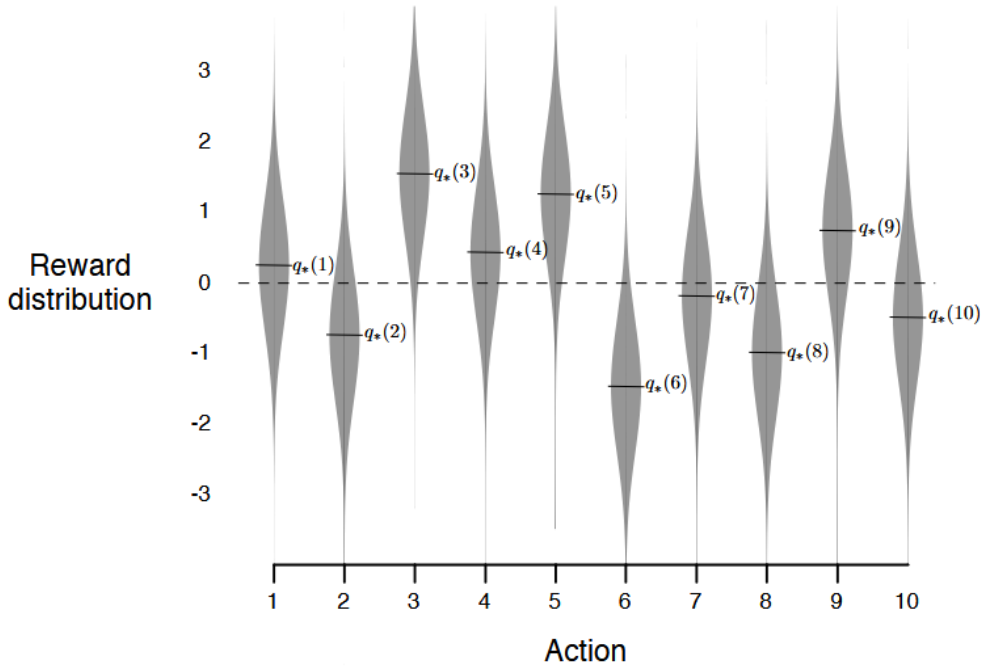
Uma vantagem desses métodos é que, no limite quando o número de passos aumenta, cada ação será amostrada um número infinito de vezes, garantindo assim que todos os Qt(a) convirjam para seus respectivos q*(a). Isso, é claro, implica que a probabilidade de selecionar a ação ótima converge para maior que 1 - ε, isto é, para quase certeza. Essas são apenas garantias assintóticas, no entanto, e dizem pouco sobre a eficácia prática dos métodos.

Exercício 2.1

Na seleção de ação ε-greedy, para o caso de duas ações e ε = 0.5, qual é a probabilidade de que a ação gananciosa (greedy action) seja selecionada?

2.3 O Testbed de 10 Braços

Para avaliar aproximadamente a eficácia relativa dos métodos de valor de ação gananciosa (greedy) e ε-greedy, os comparamos numericamente em um conjunto de problemas de teste. Este foi um conjunto de 2000 problemas de k-armed bandit gerados aleatoriamente com k = 10. Para cada problema de bandit, os valores de ação, q*(a), a = 1,..., 10, foram selecionados de acordo com uma distribuição normal com média zero e variância unitária, e então as recompensas reais foram selecionadas de acordo com uma distribuição normal com média q*(a) e variância unitária.



A Figura 2.1 mostra um exemplo de problema de bandit do testbed de 10 braços (10-armed testbed). O valor verdadeiro q*(a) de cada uma das dez ações foi selecionado de acordo com uma distribuição normal com média zero e variância unitária, e então as recompensas reais foram selecionadas de acordo com uma distribuição normal com média q*(a) e variância unitária, como sugerido por essas distribuições cinza.

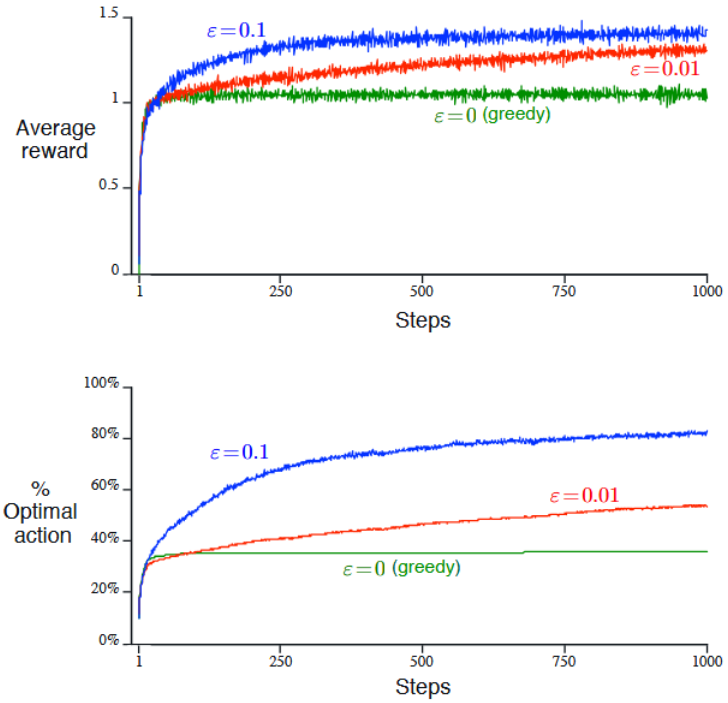


Figure 2.2: Average performance of ε-greedy action-value methods on the 10-armed testbed. These data are averages over 2000 runs with different bandit problems. All methods used sample averages as their action-value estimates.

Os resultados médios para os métodos ganancioso (greedy) e ε-greedy no testbed de 10 braços são mostrados na Figura 2.2. Essas são curvas de

aprendizado mostrando como o desempenho melhora com a experiência ao longo de 1000 passos de ação. Conforme pode ser visto, o método ganancioso (greedy) melhorou ligeiramente mais rápido que os outros no início, mas então se estabilizou em um nível inferior. Ele alcançou uma recompensa de apenas cerca de 1 após 1000 passos. O problema foi que ele ficou "preso" tentando a ação sub-ótima. A curva inferior mostra que o método ganancioso (greedy) encontrou a ação ótima em apenas aproximadamente um terço das tarefas. Nos outros dois terços, suas amostras iniciais da ação ótima foram decepcionantes, e nunca retornou a ela. Os métodos ε-greedy eventualmente tiveram melhor desempenho porque continuaram a fazer exploração (exploration) e melhorar suas chances de reconhecer a ação ótima.

Exemplo prático: Pense em um aplicativo de delivery testando diferentes algoritmos de roteamento. O método ganancioso (greedy) pode rapidamente adotar o primeiro algoritmo que parece funcionar bem, mas pode perder algoritmos potencialmente melhores. O método ε-greedy ocasionalmente testa outros algoritmos, podendo descobrir rotas mais eficientes a longo prazo.

2.4 Implementação Incremental

Os métodos de valor de ação que discutimos até agora estimam valores de ação como médias amostrais das recompensas observadas. Agora nos voltamos para a questão de como essas médias podem ser computadas de maneira computacionalmente eficiente, em particular, com memória constante e computação constante por passo de tempo.

Para manter uma média em execução de uma sequência de recompensas, parece à primeira vista que devemos manter um registro de todas as recompensas e então realizar a divisão sempre que a média seja necessária. Mas isso não é necessário. É fácil conceber métodos incrementais para computar médias amostrais que requerem memória e computação apenas para as estimativas atuais.

Dado Qn e a n-ésima recompensa, Rn, a nova média de todas as n recompensas pode ser computada por:

$$Q_{n+1} = (1/n) \times [R_1 + R_2 + \ldots + R_n] = (1/n) \times [R_n + \Sigma_{i=1 \text{ a } n-1} R_i] = (1/n) \times [R_n + (n-1) \times Q_n] = (1/n) \times [R_n + nQ_n - Q_n] = Q_n + (1/n) \times [R_n - Q_n]$$

Esta implementação requer memória apenas para Qn e n, e apenas uma pequena computação para cada nova recompensa. O método usa o tamanho do passo (step size) 1/n. Neste livro denotamos o parâmetro de tamanho do passo por α ou, mais geralmente, por αt(a).

Um pseudocódigo para um algoritmo de bandit completo usando médias amostrais computadas incrementalmente e seleção de ação ε-greedy é mostrado abaixo:

Algoritmo de bandit simples:

Inicializar, para a = 1 até k:

$Q(a) \leftarrow 0$

N(a) ← 0

Loop infinito:

A ← {argmax_a Q(a) com probabilidade 1-ε (quebrando empates aleatoriamente)

{ação aleatória com probabilidade ε

R ← bandit(A)

N(A) ← N(A) + 1

Q(A) ← Q(A) + (1/N(A)) × [R - Q(A)]

Exemplo prático: Em um sistema de recomendação de música, este algoritmo incremental permite atualizar as preferências do usuário em tempo real. Cada vez que o usuário ouve uma música completa (recompensa positiva) ou pula uma música (recompensa negativa), o sistema atualiza suas estimativas sem precisar recalcular todas as interações passadas.

2.5 Rastreando um Problema Não-Estacionário

Os métodos de média discutidos até agora são apropriados para problemas de bandit estacionários, isto é, para problemas de bandit nos quais as probabilidades de recompensa não mudam ao longo do tempo. Como notado anteriormente, frequentemente encontramos problemas de aprendizado por reforço que são efetivamente não-estacionários. Em tais casos, faz sentido dar mais peso às recompensas recentes do que às recompensas do passado distante. Uma das maneiras mais populares de fazer isso é usar um parâmetro de tamanho de passo (step-size parameter) constante.

Por exemplo, a regra de atualização incremental (2.3) para atualizar uma média Qn das n-1 recompensas passadas é modificada para ser:

Qn+1 = Qn + α × [Rn - Qn]

onde o parâmetro de tamanho de passo α ∈ (0, 1] é constante. Isso resulta em Qn+1 sendo uma média ponderada de recompensas passadas e da estimativa inicial Q1:

Qn+1 = Qn + α × [Rn - Qn] = αRn + (1-α)Qn = αRn + (1-α)[αRn-1 + (1-α)Qn-1] = αRn + (1-α)αRn-1 + (1-α)²Qn-1 = ... = (1-α)^n Q1 + Σ(i=1 a n) α(1-α)^(n-i) Ri

Chamamos isso de média ponderada por recência exponencial (exponential recency-weighted average). O peso, α(1-α)^(n-i), dado à recompensa Ri depende de quantos passos de recompensa atrás foi observada: o peso diminui exponencialmente de acordo com o expoente na quantidade (1-α)^(n-i). De acordo, isso às vezes é chamado de média com esquecimento exponencial (exponential forgetting average).

Exemplo prático: Em trading algorítmico, as condições de mercado mudam constantemente. Um sistema que usa média com esquecimento exponencial

dará mais importância aos dados de preços recentes do que aos dados antigos, adaptando-se mais rapidamente às mudanças nas tendências de mercado.

2.6 Valores Iniciais Otimistas

Todos os métodos que discutimos até agora são dependentes até certo ponto das estimativas de valor de ação iniciais, Q1(a). Na linguagem da estatística, esses métodos são enviesados por suas estimativas iniciais. Para os métodos de média amostral, o viés desaparece uma vez que todas as ações tenham sido selecionadas pelo menos uma vez, mas para métodos com α constante, o viés é permanente, embora diminua ao longo do tempo.

Na prática, esse tipo de viés geralmente não é um problema e às vezes pode ser muito útil. O lado negativo é que as estimativas iniciais se tornam, em efeito, um conjunto de parâmetros que devem ser escolhidos pelo usuário. O lado positivo é que eles fornecem uma maneira fácil de fornecer algum conhecimento prévio sobre que nível de recompensas pode ser esperado.

Valores de ação iniciais também podem ser usados como uma maneira simples de encorajar exploração (exploration). Suponha que em vez de definir os valores de ação iniciais para zero, como fizemos no testbed de 10 braços, os definimos todos para +5. Lembre-se de que os q*(a) neste problema são selecionados de uma distribuição normal com média 0 e variância 1. Uma estimativa inicial de +5 é, portanto, muito otimista.

Mas esse otimismo encoraja métodos de valor de ação a fazer exploração (exploration). Quaisquer que sejam as ações inicialmente selecionadas, a recompensa é menor que as estimativas iniciais; o aprendiz muda para outras ações, sendo "decepcionado" com as recompensas que está recebendo. O resultado é que todas as ações são tentadas várias vezes antes que as estimativas de valor convirjam. O sistema faz uma quantidade considerável de exploração (exploration) mesmo se ações gananciosas (greedy actions) são selecionadas o tempo todo.

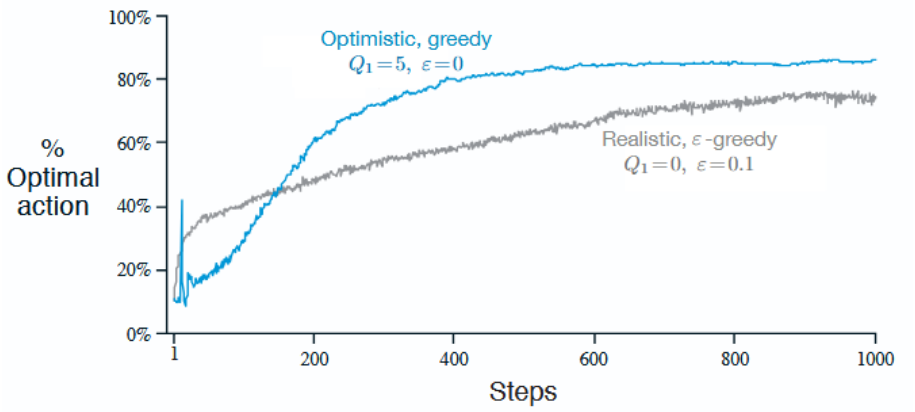


Figure 2.3: The effect of optimistic initial action-value estimates on the 10-armed testbed. Both methods used a constant step-size parameter, $\alpha = 0.1$.

A Figura 2.3 mostra o desempenho no testbed de 10 braços de um método ganancioso (greedy) usando Q1(a) = +5, para todo a. Para comparação, também é mostrado um método ε-greedy com Q1(a) = 0. Inicialmente, o método otimista tem pior desempenho porque faz mais exploração (exploration), mas

eventualmente tem melhor desempenho porque sua exploração (exploration) diminui com o tempo. Chamamos essa técnica para encorajar exploração (exploration) de valores iniciais otimistas (optimistic initial values).

Exemplo prático: Um sistema de anúncios online pode começar assumindo que todos os novos anúncios têm uma taxa de clique muito alta (valores otimistas). Isso força o sistema a testar todos os anúncios pelo menos algumas vezes antes de se concentrar nos que realmente funcionam melhor.

2.7 Seleção de Ação por Upper Confidence Bound

A exploração (exploration) é necessária porque sempre há incerteza sobre a precisão das estimativas de valor de ação. As ações gananciosas (greedy actions) são aquelas que parecem melhores no presente, mas algumas das outras ações podem realmente ser melhores. A seleção de ação ε-greedy força as ações não-gananciosas a serem tentadas, mas indiscriminadamente, sem preferência por aquelas que são quase gananciosas ou particularmente incertas. Seria melhor selecionar entre as ações não-gananciosas de acordo com seu potencial para realmente serem ótimas, levando em conta tanto quão próximas suas estimativas estão de serem máximas quanto as incertezas nessas estimativas.

Uma maneira eficaz de fazer isso é selecionar ações de acordo com:

$$A_t = \operatorname{argmax}_a \left[Q_t(a) + c \times \sqrt{\frac{\ln t}{N_t(a)}} \right]$$

onde ln t denota o logaritmo natural de t (o número que e ≈ 2.71828 teria que ser elevado para ser igual a t), Nt(a) denota o número de vezes que a ação a foi selecionada antes do tempo t (o denominador em (2.1)), e o número c > 0 controla o grau de exploração (exploration). Se Nt(a) = 0, então a é considerada uma ação maximizadora.

A ideia desta seleção de ação por Upper Confidence Bound (UCB) é que o termo da raiz quadrada é uma medida da incerteza ou variância na estimativa do valor de a. A quantidade sendo maximizada é, portanto, uma espécie de limite superior no possível valor verdadeiro da ação a, com c determinando o nível de confiança. Cada vez que a é selecionada, a incerteza é presumivelmente reduzida: Nt(a) incrementa, e, como aparece no denominador, o termo de incerteza diminui.

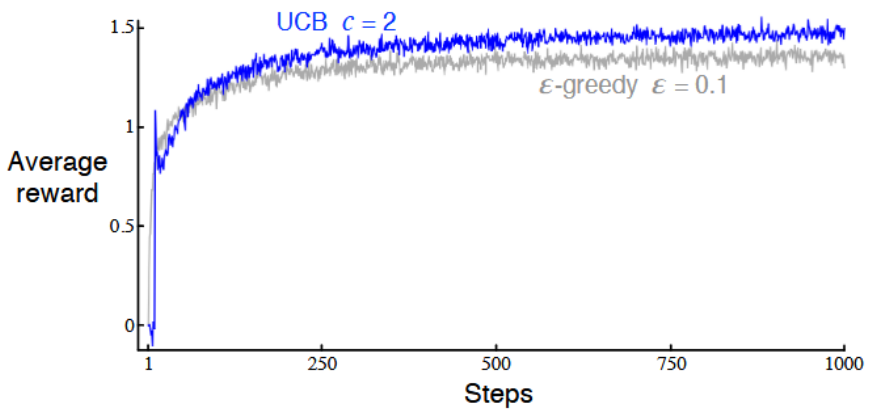


Figure 2.4: Average performance of UCB action selection on the 10-armed testbed. As shown, UCB generally performs better than ε-greedy action selection, except in the first k steps, when it selects randomly among the as-yet-untried actions.

Os resultados com UCB no testbed de 10 braços são mostrados na Figura 2.4. UCB frequentemente funciona bem, como mostrado aqui, mas é mais difícil que ε-greedy para estender além de bandits para as configurações de aprendizado por reforço mais gerais consideradas no resto deste livro.

Exemplo prático: Em ensaios clínicos médicos, o UCB pode ser usado para alocar pacientes a diferentes tratamentos. Tratamentos com poucos dados (alta incerteza) ou que parecem promissores recebem mais pacientes, mas de forma inteligente baseada tanto na eficácia estimada quanto na incerteza dessa estimativa.

2.8 Algoritmos de Gradient Bandit

Até agora neste capítulo consideramos métodos que estimam valores de ação e usam essas estimativas para selecionar ações. Esta é frequentemente uma boa abordagem, mas não é a única possível. Nesta seção consideramos aprender uma preferência numérica para cada ação a, que denotamos $H_t(a) \in \mathbb{R}$. Quanto maior a preferência, mais frequentemente essa ação é tomada, mas a preferência não tem interpretação em termos de recompensa. Apenas a preferência relativa de uma ação sobre outra é importante; se adicionamos 1000 a todas as preferências de ação, não há efeito sobre as probabilidades de ação, que são determinadas de acordo com uma distribuição soft-max (isto é, distribuição de Gibbs ou Boltzmann) como segue:

$$\Pr\{A_t = a\} = e^{H_t(a)} / \left(\sum_{b=1}^{k} e^{H_t(b)}\right) = \pi_t(a)$$

onde também introduzimos uma notação nova útil, $\pi_t(a)$, para a probabilidade de tomar a ação a no tempo t. Inicialmente, todas as preferências de ação são iguais (por exemplo, $H_1(a) = 0$, para todo a) de modo que todas as ações tenham uma probabilidade igual de serem selecionadas.

Há um algoritmo de aprendizado natural para preferências de ação soft-max baseado na ideia de ascensão de gradiente estocástico (stochastic gradient ascent). A cada passo, após selecionar a ação A_t e receber a recompensa R_t, as preferências de ação são atualizadas por:

$$H_{t+1}(A_t) = H_t(A_t) + \alpha(R_t - \bar{R}_t)(1 - \pi_t(A_t))$$

e

$$H_{t+1}(a) = H_t(a) - \alpha(R_t - \bar{R}_t)\pi_t(a), \text{ para todo } a \neq A_t$$

onde $\alpha > 0$ é um parâmetro de tamanho de passo, e $\bar{R}_t \in \mathbb{R}$ é a média das recompensas até, mas não incluindo, o tempo t. O termo $\bar{R}_t$ serve como uma baseline com a qual a recompensa é comparada. Se a recompensa for maior que a baseline, então a probabilidade de tomar A_t no futuro é aumentada, e se a recompensa for abaixo da baseline, então a probabilidade é diminuída. As ações não selecionadas se movem na direção oposta.

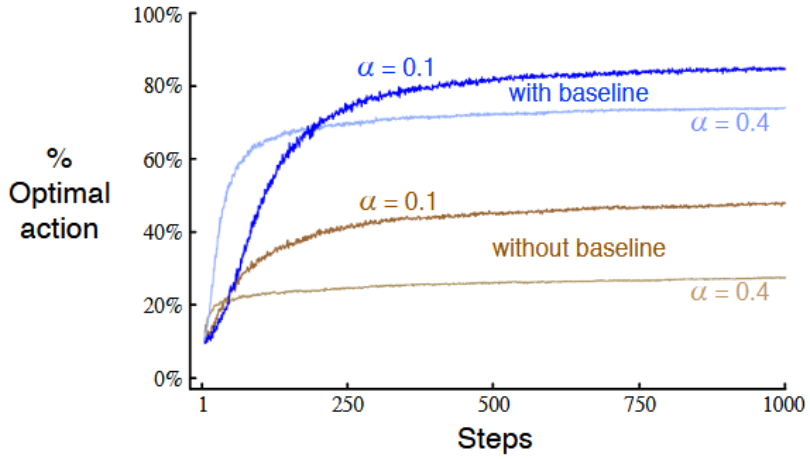


Figure 2.5: Average performance of the gradient bandit algorithm with and without a reward baseline on the 10-armed testbed when the $q_*(a)$ are chosen to be near +4 rather than near zero.

A Figura 2.5 mostra resultados com o algoritmo de gradient bandit em uma variante do testbed de 10 braços em que as recompensas esperadas verdadeiras foram selecionadas de acordo com uma distribuição normal com média de +4 em vez de zero. Este deslocamento para cima de todas as recompensas não tem absolutamente nenhum efeito sobre o algoritmo de gradient bandit porque do termo de baseline de recompensa, que instantaneamente se adapta ao novo nível.

Exemplo prático: Em sistemas de recomendação de notícias, o gradient bandit pode aprender preferências por diferentes categorias de notícias. Se artigos de tecnologia recebem mais cliques que a média (baseline), o sistema aumenta a preferência por tecnologia. Se artigos de esporte recebem menos cliques, a preferência diminui.

2.10 Resumo

Apresentamos neste capítulo várias maneiras simples de equilibrar exploração (exploration) e aproveitamento (exploitation). Os métodos ε-greedy escolhem aleatoriamente uma pequena fração do tempo, enquanto os métodos UCB escolhem deterministicamente mas conseguem exploração (exploration) favorecendo sutilmente a cada passo as ações que até agora receberam menos amostras. Algoritmos de gradient bandit estimam não valores de ação, mas preferências de ação, e favorecem as ações mais preferidas de maneira graduada e probabilística usando uma distribuição soft-max. O expediente simples de inicializar estimativas otimisticamente faz com que mesmo métodos gananciosos (greedy methods) façam exploração (exploration) significativa.

É natural perguntar qual desses métodos é melhor. Embora esta seja uma pergunta difícil de responder em geral, certamente podemos executá-los todos no testbed de 10 braços que usamos ao longo deste capítulo e comparar seus desempenhos. Uma complicação é que todos eles têm um parâmetro; para obter uma comparação significativa, temos que considerar seu desempenho como uma função de seu parâmetro.

A Figura 2.6 mostra esta medida para os vários algoritmos de bandit deste capítulo, cada um como uma função de seu próprio parâmetro mostrado em uma

única escala no eixo x. Este tipo de gráfico é chamado de estudo de parâmetro (parameter study). Note que os valores dos parâmetros são variados por fatores de dois e apresentados em uma escala logarítmica. Note também as formas características de U invertido do desempenho de cada algoritmo; todos os algoritmos têm melhor desempenho em um valor intermediário de seu parâmetro, nem muito grande nem muito pequeno.

Lições práticas: O dilema exploração (exploration) vs aproveitamento (exploitation) aparece em muitas decisões do dia a dia:

- **Escolha de restaurante**: Ir ao seu favorito (aproveitamento) ou tentar um novo (exploração)?
- **Carreira profissional**: Ficar em um emprego estável (aproveitamento) ou buscar novas oportunidades (exploração)?
- **Investimentos**: Manter portfólio conservador conhecido (aproveitamento) ou diversificar em novos ativos (exploração)?

Observações Bibliográficas e Históricas

2.1 Problemas de bandit têm sido estudados em estatística, engenharia e psicologia. Em estatística, problemas de bandit se enquadram no cabeçalho "sequential design of experiments", introduzido por Thompson (1933, 1934) e Robbins (1952), e estudado por Bellman (1956).

2.2 Métodos de valor de ação para nosso problema de k-armed bandit foram propostos primeiro por Thathachar e Sastry (1985). Estes são frequentemente chamados de estimator algorithms na literatura de learning automata. O termo action value é devido a Watkins (1989).

2.4–5 Este material se enquadra no cabeçalho geral de stochastic iterative algorithms, que é bem coberto por Bertsekas e Tsitsiklis (1996).

2.6 Optimistic initialization foi usada em aprendizado por reforço por Sutton (1996).

2.7 Trabalho inicial sobre usar estimativas do upper confidence bound para selecionar ações foi feito por Lai e Robbins (1985), Kaelbling (1993b), e Agrawal (1995). O algoritmo UCB que apresentamos aqui é chamado UCB1 na literatura e foi desenvolvido primeiro por Auer, Cesa-Bianchi e Fischer (2002).

2.8 Algoritmos de gradient bandit são um caso especial dos gradient-based reinforcement learning algorithms introduzidos por Williams (1992) que mais tarde se desenvolveram nos actor-critic e policy-gradient algorithms que tratamos mais tarde neste livro.