Introduction to Bandits

CSC_52081_EP - Lecture Notes

Jesse Read

Last updated: January 22, 2025

The main concepts to take away today:

- How is the bandits setting different from (and similar to) settings we have seen earlier (for example, multi-output prediction)? What are the main challenges?
- How to set up a bandit problem (multi-arm stochastic bandit). Possible applications.
- Notions of regret
- Upper Confidence Bounds (UCB) algorithm
- Intuition behind the Bayesian setting; and Thomspon's Sampling
- How bandits relate to reinforcement learning.

References:

- In [2] there are some nice formalisations of regret and risk, which relate to questions raised in Week 2 (multi-output prediction)
- In [4] (Chapter 2) Multi-armed Bandits.
- In [3] (Chapter 5) Concentration of Measure.

0 A Brief Aside: Sequential Monte Carlo

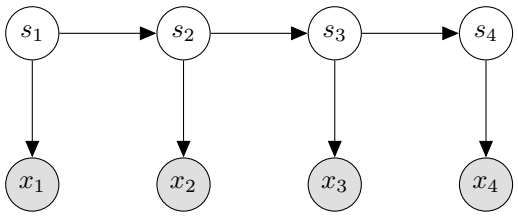

Figure 1: A Bayesian network representing joint distribution $p(s_1, \ldots, s_t, x_1, \ldots, x_t)$.

Suppose we want

$$\mathbb{E}_{S_t \sim P(S_t|x_{1:t})}[r(S_t)|x_{1:t}]$$

where $r(S_t) \equiv r(S_t, a) = [\![S_t = a]\!]$ (an indicator function). We develop as follows, noting the model given in Fig. 1:

$$
\begin{aligned}
p(s_t, x_{1:t}) &= \sum_{s_1,\ldots,s_{t-1}} p(x_t|s_t)p(s_t|s_{t-1}) \cdots p(x_1|s_1)p(s_1|s_0) \\
\mathbb{E}_{S_t \sim P(S_t|x_{1:t})}[r(S_t)|x_t] &= \frac{1}{Z} \sum_{s_t} r(s_t) \cdot p(s_t, x_{1:t}) \quad \triangleright \text{ because } p(s_t|x_{1:t}) = \frac{1}{Z} p(s_t, x_{1:t}) \\
&= \mathbb{E}_{S_t \sim Q} \left[\frac{r(s_t) \cdot \overbrace{p(x_t|s_t)p(s_t|s_{t-1}) \cdot \omega_{t-1}}^{\omega_t}}{q(s_t)} \right] \quad \triangleright \text{ Importance Sampling} \\
&= \mathbb{E}_{S_t \sim Q} \left[\frac{r(s_t) \cdot p(x_t|s_t) \cancel{p(s_t|s_{t-1})} \cdot \omega_{t-1}}{\cancel{p(s_t|s_{t-1})}} \right] \quad \text{where } q(s_t) \equiv p(s_t|s_{t-1}) \\
&= \mathbb{E}_{S_t \sim Q} \left[r(s_t) \cdot p(x_t|s_t) \cdot \omega_{t-1} \right] \\
&= \underbrace{p(x_t|s_t) \cdot \omega_{t-1}}_{\omega_t} \quad \triangleright \text{ because } \mathbb{E}[[\![s = a]\!]] = P(s = a)
\end{aligned}
$$

So, for $i = 1, \ldots, M$, we progressively sample $s_t^{(i)} \sim p(s_t|s_{t-1}^{(i)})$ and weight it by $\omega_t^{(i)}$, for $t = 1, 2, \ldots, T$. We should not forget about $\frac{1}{Z}$; indeed we should normalize $\widetilde{\omega}^{(i)}$ so they sum to 1 (a valid probability distribution). The distribution we are talking about is the one we are were looking for:

$$
P(s_t = a \mid x_{1:t}) \approx \sum_{i=1}^{M} [\![s_t^{(i)} = a]\!] \cdot \omega_t^{(i)} \tag{1}
$$

To recover $P(s_{1:t} \mid x_{1:t})$ (if necessary), we need to track particle paths (products through time), and then normalize over all particles at the end. Namely, keep track of the path $P(s_1, \ldots, s_t) = \omega_{\texttt{path}}^{(i)} = \omega_1^{(i)} \cdot \omega_2^{(i)} \cdot \cdots \cdot \omega_t^{(i)}$ then normalise $\widetilde{\omega}_{\texttt{path}}^{(1)}, \ldots, \widetilde{\omega}_{\texttt{path}}^{(m)}$.

Implementation Sketch

For M 'particles', sample from the initial state:

$$
\begin{aligned}
s_1^{(i)} &\sim p(s_1) \\
w^{(i)} &= \frac{1}{M}
\end{aligned}
$$

Then for $t = 1, \ldots, T$, we sample from the proposal distribution

$$
s_t^{(i)} \sim p(s_t \mid s_{t-1}^{(i)})
$$

and we apply the importance weight

$$
\omega_t^{(i)} \propto p(x_t \mid s_t^{(i)}) \cdot \omega_{t-1}^{(i)}
$$

at some point we need to apply our normalization constant (to have a valid distribution),

$$
\widetilde{\omega}_t^{(i)} = \frac{1}{Z} \omega_t^{(i)}
$$

where Z is the sum over all particle weights.

Intuitively: the probability that $s_t = a$ is the sum over all particles where $s_t = a$ times their weight; as per Eq. (1).

Remarks

Particle filters normally include a resampling step, which we have omitted here (due to the relatively short trajectories).

Particles are ideal for tracking and localization in $\boldsymbol{s}_t \in \mathbb{R}^d$.

Main takeaway: sometimes sampling from p is difficult, we sample from q instead, and (given the correct importance weight) this can be a valid thing to do.

1 Recap and Intro (from Multi-Label Prediction to Bandits)

In multi-label prediction, we predict T labels (we talked about m labels, but we use T here to make a closer analogy/contrast to bandits):

$$h(\boldsymbol{x}) = \widehat{\boldsymbol{y}} = [\widehat{y}_1, \ldots, \widehat{y}_T]$$

and gain some reward $r(\boldsymbol{y}, \widehat{\boldsymbol{y}})$ for this prediction. The issue with bandits is that $\widehat{y}_t$ (that we consider as an action) is required *before* $\widehat{y}_{t+1}$. In this sense, because of the strict time order of decisions, we may experience regret at time $t+1$, respective of our earlier decisions. Our goal, at any time t, is to minimize expected regret.

Figure 2: Is it possible to get zero regret (*regret nothing*), like Edith Piaf?

1.1 Expected Regret

The expected regret is the difference between what we expect in terms of reward from making the optimal decision(s) and what we expect from *our* decision(s). Minimizing loss is equivalent to maximizing reward. Minimizing regret is the same as minimizing loss, but regret is not the same as loss; namely, it is always possible to get 0 regret by acting optimally, but optimal actions do not always result in 0 loss.

Regret in multi-label prediction

Our regret, using model h for a prediction over T labels for instance $\boldsymbol{x}$:

$$\mathcal{R}_T(h, \boldsymbol{x}) = \underbrace{\mathbb{E}_{\boldsymbol{Y} \sim P(\boldsymbol{Y}|\boldsymbol{x})}(r(\boldsymbol{Y}, \boldsymbol{y}^*))}_{\text{optimal}} - \underbrace{\mathbb{E}_{\boldsymbol{Y} \sim P(\boldsymbol{Y}|\boldsymbol{x})}(r(\boldsymbol{Y}, h(\boldsymbol{x})))}_{\text{what you did (model } h\text{)}} \tag{2}$$

where *Bayes-optimal* decision $\boldsymbol{y}^*$ (assume $P \approx P_\star$) wrt reward r (higher is better). The randomness (uncertainty) modeled by the expectation is due to not knowing the true label.

Suppose Hamming score, then we have

$$\mathbb{E}_{\boldsymbol{Y}\sim P(\boldsymbol{Y}|\boldsymbol{x})}\left[r(\boldsymbol{Y},\boldsymbol{y}^*)\right] = \sum_{\boldsymbol{y}\in\{0,1\}^T} p(\boldsymbol{y}\mid\boldsymbol{x})r(\boldsymbol{y},\widehat{\boldsymbol{y}})$$
$$= \sum_{\boldsymbol{y}\in\{0,1\}^m} \underbrace{\left[\sum_{t=1}^{T}[\![y_t = y_t^*]\!]\right]}_{r(\boldsymbol{y},\widehat{\boldsymbol{y}})} P(\boldsymbol{y}|\boldsymbol{x})$$

Since $\boldsymbol{y}^* \in \{0,1\}^m$, and since all decision making regarding $y_1,\ldots,y_L$ is completed prior to taking an action, it is possible to get 0 regret with full exploration of $\{0,1\}^m$. A tradeoff exists between exploration vs computational complexity.

Note how Hamming loss is a kind of cumulative reward $\sum_{t=1}^{T} R_t$.

Regret in bandits

At time-step T, our expected regret:

$$\mathcal{R}_T(\pi,\nu) = \mu^* T - \mathbb{E}_{A\sim\pi,R\sim\nu(R|A)}\left[\sum_{t=1}^{T} R_t\right]$$

where μ^* is the mean [reward] of the optimal arm. This is not so different from Eq. (2). Here, the randomness stems from the stochastic nature of the bandit (distribution $R_t \sim \nu_a$), which happens because we marginalize out the state, and the policy $A \sim \pi$ behind the choice of arm, at each step t (analogous to the classifier h).

1.2 Bandits as reinforcement learning with the state marginalized out

In bandits there is no state (s_t) or observation ($\boldsymbol{x}_t$). See Fig. 3.

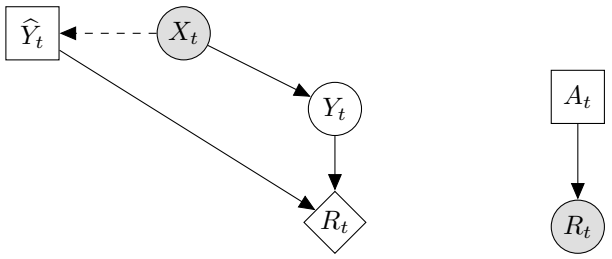

Figure 3: Influence diagrams (recall: action/decision nodes are square). We are already familiar with the decision process representing prediction (left). In bandits (right) there is no observation. We can either suppose that an unobserved state-node S_t pointing towards R_t has been marginalized (thus, R_t inheriting its randomness) or than R_t is itself stochastic. If there were no randomness, we are no longer talking about stochastic bandits.

Whereas, previously in prediction, we had observation nodes hinting towards the current state, in bandits, there is no state. We can imagine that it has been marginalised out:

$$q_*(a) = \mathbb{E}_{R_t\sim\nu_a}[R_t|A_t = a] = \sum_{s\in\mathcal{S}} r(s,a)P(S_t = s) \tag{3}$$

such that, even if $r(s,a)$ is deterministic, we may still consider that reward R is a random variable that inherits the randomness of S.

1.3 Exploration vs exploitation (bandits)

In bandits: the challenge is: arm/action/decision A_t (analogious to $\widehat{Y}_t$ in multi-output prediction) is required *at time t*. This leads to tradeoff between exploration vs exploitation; we cannot do both. By pulling an arm we generate training pair (a_t, r_t), which aggregates to our knowledge $\{(a_\tau, r_\tau)\}_{\tau=1}^t$. We can either *exploit* that knowledge (by pulling the best arm according to our knowledge) or *explore* further (by pulling any arm).

The fact that we are never given the true a_t^* as a 'training label' (as we are given y_t in supervised learning) is an important aspect that differentiates bandits (and reinforcement learning) from supervised learning.

2 Tail Probabilities and Bounds

We are interested in learning about $\nu_1, \ldots, \nu_k$ (the distributions of rewards of each arm; recall Eq. (3)), and in particular the means of these distributions $\mu_1, \ldots, \mu_k$; because an optimal agent will take the action/pull the arm with the highest expected reward/highest mean ($q_*(a) = \mu_a$).

Suppose iid samples from arm a (Warning: Change in notation! X_t will be our generic variable of interest in this section – we de not discuss bandits in particular): $X_1, X_2, \ldots, X_T$. Our unbiased estimate (empirical mean over T draws):

$$\hat{\mu} = \frac{1}{T}\sum_{t=1}^{T} X_t$$

with

$$\mathbb{E}[\hat{\mu}] = \mu$$
$$\mathbb{V}[\hat{\mu}] = \mathbb{E}[(\hat{\mu} - \mu)^2] = \frac{\sigma^2}{T}.$$

A pertinent question: How confident can we be that the true mean μ is not outside the range $\mu \pm \epsilon$? In other words, queries $\mathbb{P}(\hat{\mu} \geq \mu + \epsilon)$ and $\mathbb{P}(\hat{\mu} \geq \mu - \epsilon)$.

What we are referring to are tail probabilities (specifically, a two-sided tail probability – because it refers to the tail of both sides); see Fig. 4. And what we are looking for is a bound.

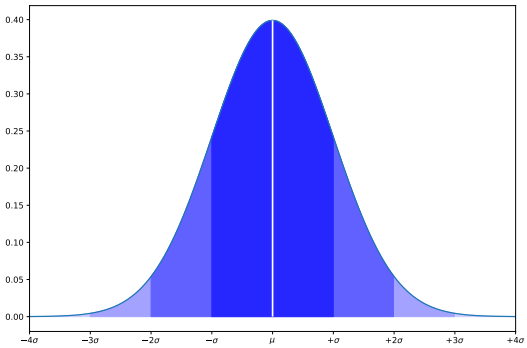

Figure 4: For any given ϵ, we can ask what is the probability mass in the tails, above $\mu + \epsilon$, and below $\mu - \epsilon$.

From Markov's inequality, $\mathbb{P}(|X| \geq \epsilon) \leq \frac{\mathbb{E}[|X|]}{\epsilon}$ we can derive (by plugging in the variable $Z = \mathbb{V}[X]$; see [3] for more rigorous development) Chebyshev's inequality:

$$P(|X - \mathbb{E}[X]| \geq \epsilon) \leq \frac{\mathbb{V}[X]}{\epsilon^2}$$

We can thus bound the two-sided tail:

$$\mathbb{P}(|\hat{\mu} - \mu| \geq \epsilon) \leq \frac{\sigma^2}{T\epsilon^2}$$

but this is a really loose bound (if not immediately obvious, try plugging in some numbers).

Hence the motivation for further development of this question.

3 Upper Confidence Bounds (UCB)

We arrive (see, e.g., [3], for full working), we arrive at the Upper Confidence Bound (UCB); Eq. (4)[1].

$$\mathsf{UCB}_{t-1}(a) = Q_{t-1}(a) + c\sqrt{\frac{\ln t}{N_t(a)}} = Q_{t-1}(a) + \sqrt{\frac{\alpha \ln t}{N_t(a)}} \tag{4}$$

(note that $\alpha = c^2$). We use c to keep a balance between exploitation (the Q_t-term) and exploration (the $\sqrt{}$ term).

Significance: we achieve logarithmic bounds on regret; which may be considered as a solution to the bandit problem (i.e., an effective balance between exploration and exploitation).

4 Bayesian View of the Bandits Problem

Bayesians[2] consider parameter(s) θ as a random variable (and thus has a distribution P), and observations r_t are fixed constants. The rules of Bayesian network still apply (see Fig. 5):

$$P(\theta, \{r_1, \ldots, r_T\}) = P(\{r_1, \ldots, r_T\} \mid \theta) P(\theta) \tag{5}$$

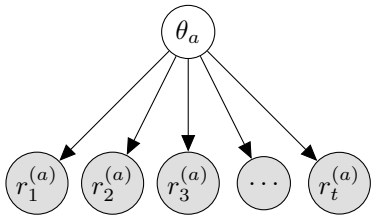

Figure 5: Bayesian network representation of Eq. (5) (for a given arm).

In other words, what is P?

Bayesian regret (note: the role of the policy π, deciding which action to choose):

$$\mathcal{R}_T(\pi, \theta) := \mathbb{E}_{\theta \sim P, R \sim \pi_\theta}[R(T)] = \mathbb{E}_{\theta \sim P}\left[\mu^* T - \sum_{t=1}^{T} \mu(a_t)\right]$$

[1] Specifically, we display UCB1 [1] (note: we must assume rewards $r \in [0, 1]$)

[2] Those that consider themselves to do Bayesian machine learning or Bayesian statistics; but note the use of Bayes' rule (as so far in the course) does not imply a Bayesian approach; 'frequentists' also use Bayes' rule

5 From Bandits to Reinforcement Learning

Bandits are an interesting problem with many real-world applications. We can also view bandits as a stepping stone on the way to reinforcement learning (RL). In RL, just like in Bandits, our action has an effect on future observations; but in RL we specifically model the change in state (i.e., direct effect we (the agent, its policy) has on the world/environment).

Consider the illustration of Monte Carlo Tree Search (MCTS) in Fig. 5; a game of Tic-Tac-Toe *in a given state* (root node): we need to choose the next action. We can sample a number of trajectories/playouts and record the reward (we may consider this $R \sim \nu_a$), with which to estimate the value of action a (we can consider this $Q_a \approx \mathbb{E}[R \mid A = a]$).

We have already seen a mechanism to do this (Monte Carlo search in a tree). Monte Carlo search in a tree is a special case of MCTS, where we only consider the simulation step ('rollout policy'). We can also choose UCB (or other bandit option) to choose the action – and the vast majority of MCTS implementations do this!

MCTS has an additional step of using the tree policy to take an action (and thus change the state of the environment). That takes us into reinforcement learning.

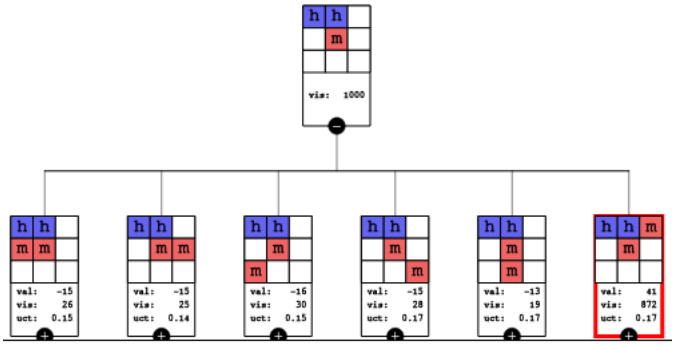

Figure 6: Found at: https://vgarciasc.github.io/mcts-viz/

References

[1] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256, 2002.

[2] Krzysztof Dembczyński, Willem Waegeman, Weiwei Cheng, and Eyke Hüllermeier. On label dependence and loss minimization in multi-label classification. *Machine Learning*, 88(1):5–45, 2012.

[3] Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020. https://tor-lattimore.com/downloads/book/book.pdf.

[4] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction, 2nd Edition*. MIT press, 2020. https://www.andrew.cmu.edu/course/10-703/textbook/BartoSutton.pdf.