18 Contextual Bandits

In many bandit problems the learner has access to additional information at the beginning of each round. Consider the problem of designing a movie recommendation system. Clearly it would be inadvisable to ignore demographic information about the user making the request, or any other contextual history such as previously watched movies or ratings. None of the algorithms presented so far can take this kind of additional information into account and the benchmark (regret) also does not measure performance relative to other sources of information. Imagine the results of trying to find the best single movie in hindsight for all users. In this chapter we will present an augmented framework and regret definition that better models the many real-world problems where contextual information is available.

⚠ Whenever you design a new benchmark, there are several factors to consider. Competing with a poor benchmark does not make sense, since even an algorithm that perfectly matches the benchmark will perform poorly. At the same time, competing with a better benchmark can be harder from a learning point of view and this penalty must be offset against the benefits.

The tradeoff just described is fundamental to all machine learning problems. In statistical estimation, the analogue tradeoff is known as the **bias-variance tradeoff**. We will not attempt to answer the question of how to resolve this tradeoff in this chapter because first we need to see how to effectively compete with improved benchmarks.

18.1 Contextual bandits: one bandit per context

In a contextual bandit problem everything works the same as in a bandit problem except the learner receives a context at the beginning of each round. The hope is that specializing the action to the context can help collect more reward. While contextual bandits can be studied in both the adversarial and stochastic frameworks, in this chapter we focus on the adversarial model. To remind the reader of the notation, let $x_{tk} \in [0, 1]$ be the reward for arm k in round t, which is chosen in advance by the adversary. The interaction model is what you would expect:

For rounds $t = 1, 2, \dots, n$:

1 Learner observes context $c_t \in \mathcal{C}$ where $\mathcal{C}$ is an arbitrary fixed set of contexts.
2 Learner selects distribution P_t on $[K]$ and samples A_t from P_t.
3 Learner observes reward $X_t = x_{tA_t}$.

A natural way to define the regret is by

$$R_n = \mathbb{E}\left[\sum_{c \in \mathcal{C}} \max_{k \in [K]} \sum_{t \in [n] : c_t = c} (x_{tk} - X_t)\right] . \tag{18.1}$$

The difference is that now we are trying to compete with the best context-dependent policy in hindsight, rather than the best fixed action. If the set of possible contexts is finite, then a simple approach is to use a separate instance of Exp3 for each context. Let

$$R_{nc} = \mathbb{E}\left[\max_{k \in [K]} \sum_{t \in [n] : c_t = c} (x_{tk} - X_t)\right]$$

be the regret due to context $c \in \mathcal{C}$. When using a separate instance of Exp3 for each context we can use the results of Chapter 11 to bound

$$R_{nc} \leq 2\sqrt{K \sum_{t=1}^n \mathbb{I}\left\{c_t = c\right\} \log(K)} , \tag{18.2}$$

where the sum inside the square root counts the number of times context $c \in \mathcal{C}$ is observed. Because this is not known in advance it is important to use an anytime version of Exp3 for which the above regret bound holds without needing to tune a learning rate that depends on the number of times the context is observed (see Exercise 28.9). Substituting (18.2) into the regret leads to

$$R_n = \sum_{c \in \mathcal{C}} R_{nc} \leq 2 \sum_{c \in \mathcal{C}} \sqrt{K \log(K) \sum_{t=1}^n \mathbb{I}\left\{c_t = c\right\}} . \tag{18.3}$$

The magnitude of the right-hand side depends on the distribution of observed contexts. On one extreme there is only one observed context and the bound is the same as the standard finite-armed bandit problem. The other extreme occurs when all contexts are observed equally often, in which case we have

$$R_n \leq 2\sqrt{nK|\mathcal{C}|\log(K)} . \tag{18.4}$$

Jensen's inequality applied to Eq. (18.3) shows that this really is the worst case (Exercise 18.1).

⚠ It is important to emphasize that the regret in Eq. (18.4) is different than the regret studied in Chapter 11. If we ignore the context and run the standard Exp3 algorithm, then we would have

$$\mathbb{E}\left[\sum_{t=1}^n X_t\right] \geq \max_{i\in[K]} \sum_{t=1}^n x_{ti} - 2\sqrt{Kn\log(K)}\,.$$

Using one version of Exp3 per context leads to

$$\mathbb{E}\left[\sum_{t=1}^n X_t\right] \geq \sum_{c\in\mathcal{C}} \max_{i\in[K]} \sum_{t\in[n]:c_t=c} x_{ti} - 2\sqrt{Kn|\mathcal{C}|\log(K)}\,.$$

Which of these bounds is preferable depends on the magnitude of n and how useful the context is. When n is very large the second bound is more likely to be preferable. On the other hand, the second bound is completely vacuous when $n \leq 4K|\mathcal{C}|\log(K)$.

18.2 Bandits with expert advice

For large context sets using one bandit algorithm per context will almost always be a poor choice because the additional precision is wasted unless the amount of data is enormous. Fortunately, however, it is seldom the case that the context set is both large and unstructured. To illustrate a common situation we return to the movie recommendation theme where the actions are movies and the context contains user information such as age, gender and recent movie preferences. In this case the context space is combinatorially large, but there is clearly a significant amount of structure inherited from the fact that the space of movies is highly structured and users in similar demographics are more likely to have similar preferences.

Another way to write Eq. (18.1) is to let Φ be the set of all functions from $\mathcal{C} \to [K]$. Then

$$R_n = \mathbb{E}\left[\max_{\phi\in\Phi} \sum_{t=1}^n (x_{t\phi(c_t)} - X_t)\right]\,. \tag{18.5}$$

The discussion above suggests we might prefer to choose Φ to be a slightly smaller set. There are many ways to do this, some of which we describe below:

Partitions

Let $\mathcal{P} \subset 2^{\mathcal{C}}$ be a partition of $\mathcal{C}$, which means that sets in $\mathcal{P}$ are disjoint and $\cup_{P\in\mathcal{P}} P = \mathcal{C}$. Then define Φ to be the set of functions from $\mathcal{C}$ to $[K]$ that are constant on each partition of $\mathcal{P}$. In this case we can run a version of Exp3 for each partition, which means the regret depends on the number of parts $|\mathcal{P}|$ rather than on the number of contexts.

Similarity functions

Let $s : \mathcal{C} \times \mathcal{C} \to [0, 1]$ be a function that we think of as measuring the similarity between pairs of contexts on the $[0, 1]$-scale. Then let Φ be the set of functions $\phi : \mathcal{C} \to [K]$ such that the average dissimilarity

$$\frac{1}{|\mathcal{C}|^2} \sum_{c,d \in \mathcal{C}} (1 - s(c, d))\mathbb{I}\{\phi(c) \neq \phi(d)\}$$

is below a user-tuned threshold $\theta \in (0, 1)$. It is not clear anymore that we can control the regret (18.5) using some simple meta algorithm on Exp3, but keeping the regret small is still a meaningful objective.

From supervised learning to bandits with expert advice

Yet another option is to run your favorite supervised learning method, training on batch data to find a collection of predictors $\phi_1, \dots, \phi_M : \mathcal{C} \to [K]$. Then we could use a bandit algorithm to compete with the best of these in an online fashion. This has the advantage that the offline training procedure can bring in the power of batch data and the whole army of supervised learning, without relying on potentially inaccurate evaluation methods that aim to pick the best of the pack. And why pick if one does not need to?

The possibilities are endless, but in any case, we would end up with a set of functions Φ with the goal of competing with the best of them. This suggests the idea that perhaps we should think more generally about some subset Φ of functions without necessarily considering the internal structure of Φ. This is the viewpoint that we will take. In fact, we will bring this one step further by noticing that once Φ has been chosen the contexts themselves play very little role. All we need in each round is the output of each function. This leads to a setting called **bandits with expert advice**.

In this model there are M experts. At the beginning of each round the experts announce their predictions of which actions are the most promising. For the sake of generality, we allow the experts to report not only a single prediction, but a probability distribution over the actions. The interpretation of this probability distribution is that the expert, if the decision was left to them, would choose the action for the round at random from the probability distribution it reported. As discussed before, in an adversarial setting it is natural to consider randomized algorithms, hence one should not be too surprised that the experts are also allowed to randomize. An application to an important practical problem is illustrated in Fig. 18.1.

The predictions of the M experts in round t is represented by a matrix $E^{(t)} \in [0, 1]^{M \times K}$ where the mth row $E^{(t)}_m$, a probability distribution of K, is the recommendation of expert m for round t. The learner and the environment, including the expert interact as follows:

Figure 18.1 Prediction with expert advice. The experts, upon seeing a foot give expert advice on what socks should fit it best. If the owner of the foot is happy, the recommendation system earns a cookie!

For rounds $t = 1, 2, 3, \dots, n$:

1 Learner observes predictions of all experts, $E^{(t)}$.
2 Learner selects a distribution P_t on $[K]$ in some way.
3 Action A_t is sampled from P_t and the reward is $X_t = x_{tA_t}$.

The regret of the learner is with respect to the total expected reward of the best *expert*:

$$R_n = \mathbb{E}\left[\max_{m \in [M]} \sum_{t=1}^n E_m^{(t)} x_t - \sum_{t=1}^n X_t\right] . \tag{18.6}$$

There is a delicate choice to be made about whether to allow the experts predictions of the experts to depend on the actions of the learner. Or whether they should be fixed from the beginning of the game in an oblivious manner. While the framework does allow learning experts, the regret definition above is not really meaningful in this case because the total reward of any of the experts will also depend on the actions chosen by the learner (through $E^{(t)}$) and in this case a more meaningful benchmark is to compare with the total reward of the experts computed under the assumption that the learner chooses some fixed action all the time. Chapter 37 will consider this type of regret for a specific problem class. However, in this chapter we restrict ourselves to non-learning, oblivious experts.

```
Input: $n$, $K$, $M$, $\eta$, $\gamma$
Set $Q_1 = (1/M, \dots, 1/M) \in [0,1]^{1\times M}$ (a row vector)
for $t = 1, \dots, n$ do
    Receive advice $E^{(t)}$
    Choose the action $A_t \sim P_t$, where $P_t = Q_t E^{(t)}$
    Receive the reward $X_t = x_{tA_t}$
    Estimate the action rewards: $\hat{X}_{ti} = 1 - \frac{\mathbb{I}\{A_t=i\}}{P_{ti}+\gamma}(1-X_t)$
    Propagate the rewards to the experts: $\tilde{X}_t = E^{(t)}\hat{X}_t$
    Update the distribution $Q_t$ using exponential weighting:
        $$Q_{t+1,i} = \frac{\exp(\eta\tilde{X}_{ti})Q_{ti}}{\sum_j \exp(\eta\tilde{X}_{tj})Q_{tj}} \quad \text{for all } i \in [M]$$
end for
```

Algorithm 10: Exp4 algorithm

18.3 Can it go higher? Exp4

Exp4 is not just an increased version number, but stands for **E**xponential weighting for **E**xploration and **E**xploitation with **E**xperts. The idea of the algorithm is very simple. Since exponential weighting worked so well in the standard bandit problem we should adopt it to the problem at hand. However, since the goal is to compete with the best expert in hindsight, it is not the actions that we should score, but the experts. The algorithm maintains a probability distribution Q_t over experts and use this to come up with the next action. Once the action is chosen, we use our favorite reward estimation procedure to estimate the rewards for all the actions, which is then used to estimate how much total reward the individual experts would have made so far. The reward estimates are then used to update Q_t using exponential weighting. The pseudocode of the algorithm is given in Algorithm 10.

Note that A_t can be chosen in two steps, first sampling $M_t \in [M]$ from Q_t and then choosing $A_t \in [K]$ from $E^{(t)}_{M_t,\cdot}$. The reader can verify that (given the past) the probability distribution of the so-selected action is also P_t. The algorithm uses $O(M)$ memory and $O(MK)$ computation per round. Hence it is only practical when M and K are reasonable.

18.4 Regret analysis

We restrict our attention to the case when $\gamma = 0$, which is the original algorithm. The version where $\gamma > 0$ is called Exp4-IX and its analysis is left to the reader in Exercise 18.3.

THEOREM 18.1 *Let $\gamma = 0$ and $\eta = \sqrt{2\log(M)/(nK)}$ and denote by R_n the* *expected regret of Exp4 defined in Algorithm 10 after n rounds. Assume that the experts are deterministic and oblivious. Then,*

$$R_n \le \sqrt{2nK\log(M)}\,. \tag{18.7}$$

The proof will use the following lemma:

LEMMA 18.1 *For any $m^* \in [M]$ it holds that*

$$\sum_{t=1}^n \tilde{X}_{tm^*} - \sum_{t=1}^n \sum_{m=1}^M Q_{tm}\tilde{X}_{tm} \le \frac{\log(M)}{\eta} + \frac{\eta}{2}\sum_{t=1}^n\sum_{m=1}^M P_{tm}(1-\hat{X}_{tm})^2\,.$$

After translating the notation, the proof of Lemma 18.1 can be extracted from the analysis of Exp3 in the proof of Theorem 11.2, a task that we leave to the reader in Exercise 18.2.

Proof of Theorem 18.1 Let m^* be the index of the best performing expert in hindsight:

$$m^* = \text{argmax}_{m\in[M]} \sum_{t=1}^n E_m^{(t)} x_t\,. \tag{18.8}$$

Applying Lemma 18.1 shows that

$$\sum_{t=1}^n \tilde{X}_{tm^*} - \sum_{t=1}^n \sum_{m=1}^M Q_{tm}\tilde{X}_{tm} \le \frac{\log(M)}{\eta} + \frac{\eta}{2}\sum_{t=1}^M\sum_{m=1}^M Q_{tm}(1-\tilde{X}_{tm})^2\,. \tag{18.9}$$

Let $\mathcal{F}_t = \sigma(E^{(1)}, A_1, E^{(2)}, A_2, \dots, A_{t-1}, E^{(t)})$, and introduce $\mathbb{E}_t[\cdot] = \mathbb{E}[\cdot|\mathcal{F}_t]$. When $\gamma = 0$ the estimator $\hat{X}_{ti}$ is unbiased so that $\mathbb{E}_t[\hat{X}_t] = x_t$ and

$$\mathbb{E}_t[\tilde{X}_t] = \mathbb{E}_t[E^{(t)}\hat{X}_t] = E^{(t)}\mathbb{E}[\hat{X}_t] = E^{(t)}x_t\,. \tag{18.10}$$

Since Q_t is $\mathcal{F}_t$-measurable, using the tower rule for conditional expectation, taking expectations of both sides of Eq. (18.9) we get

$$R_n \le \frac{\log(M)}{\eta} + \frac{\eta}{2}\sum_{t=1}^n\sum_{m=1}^M \mathbb{E}\left[Q_{tm}(1-\tilde{X}_{tm})^2\right]\,, \tag{18.11}$$

where we also used the assumption that the experts are oblivious, and hence m^* is non-random.

Like in Chapter 11, losses are more convenient than rewards to work with. Let $\hat{Y}_{ti} = 1 - \hat{X}_{ti}$, $y_{ti} = 1 - x_{ti}$ and $\tilde{Y}_{tm} = 1 - \tilde{X}_{tm}$. Note that $\tilde{Y}_t = E^{(t)}\hat{Y}_t$ and recall also the notation $A_{ti} = \mathbb{I}\{A_t = i\}$, which means that $\hat{Y}_{ti} = \frac{A_{ti}y_{ti}}{P_{ti}}$ and

$$\mathbb{E}_t[\tilde{Y}_{tm}^2] = \mathbb{E}_t\left[\left(\frac{E_{mA_t}^{(t)} y_{tA_t}}{P_{tA_t}}\right)^2\right] = \sum_{i=1}^K \frac{\left(E_{mi}^{(t)}y_{ti}\right)^2}{P_{ti}} \le \sum_{i=1}^K \frac{E_{mi}^{(t)}}{P_{ti}}\,. \tag{18.12}$$

Therefore using the definition of P_{ti},

$$\mathbb{E}\left[\sum_{m=1}^{M} Q_{tm}(1-\tilde{X}_{tm})^2\right] \leq \mathbb{E}\left[\sum_{m=1}^{M} Q_{tm} \sum_{i=1}^{K} \frac{E^{(t)}_{mi}}{P_{ti}}\right]$$

$$= \mathbb{E}\left[\sum_{i=1}^{K} \frac{\sum_{m=1}^{M} Q_{tm} E^{(t)}_{mi}}{P_{ti}}\right] = K\,.$$

Substituting into Eq. (18.11) leads to

$$R_n \leq \frac{\log(M)}{\eta} + \frac{\eta n K}{2} = \sqrt{2nK\log(M)}\,. \qquad \square$$

Let us see how this theorem can be applied to the contextual bandit where $\mathcal{C}$ is a finite set and Φ is the set of all functions from $\mathcal{C} \to [K]$. To each of these functions $\phi \in \Phi$ we associate an expert m with $E^{(t)}_{mi} = \mathbb{I}\{\phi(c_t) = i\}$. Then $M = K^{\mathcal{C}}$ and Theorem 18.1 says that

$$R_n \leq \sqrt{2nK|\mathcal{C}|\log(K)}\,,$$

which is the same bound we derived using an independent copy of Exp3 for each context. More generally, if $\mathcal{C}$ is arbitrary (possibly infinite) and Φ is a finite set of functions from $\mathcal{C}$ to $[K]$, then the theorem ensures that

$$R_n \leq \sqrt{2nK\log(|\Phi|)}\,.$$

These results seem quite promising already, but in fact there is another improvement possible. Define random variable E^*_t by

$$E^*_t = \sum_{s=1}^{t}\sum_{i=1}^{K} \max_{m\in[M]} E^{(s)}_{mi}\,.$$

By modifying the algorithm to use an adaptive learning rate of $\eta_t = \sqrt{\log(M)/E^*_t}$ one can prove the following theorem.

THEOREM 18.2 *Assume the same conditions as in Theorem 18.1, except that let* $\eta_t = \sqrt{\log(M)/E^*_t}$. *Then, there exists a universal constant* $C > 0$ *such that*

$$R_n \leq C\sqrt{E^*_n \log(M)}\,.$$

The proof of this result is not hard and is left to the reader in Exercise 18.4. The bound on the right-hand side of the above inequality is *data-dependent* since it depends on E^*_n. It is not hard to see (Exercise 18.7) that

$$E^*_n \leq n\min(K, M) \tag{18.13}$$

and as such this bound is much better than the bound of Theorem 18.1 when $M \leq K$. One can think of E^*_n/n as the effective number of experts which depends on the degree of disagreement, or diversity in the experts' recommendations. The bound tells us that Exp4 with the suggested learning rate is able to *adapt* to the degree of disagreement between the experts. In fact, it is reasonable that learning becomes easier (and the regret bound will be smaller) when the experts tend to agree. At the same time, what is the use of having many experts if they tend to agree? This is another manifestation of the bias-variance tradeoff mentioned at the beginning. Regardless, the fact that Exp4 with an adaptive learning rate learns faster when there is more agreement amongst the experts is reassuring.

18.5 Notes

1 Perhaps the most important point of this chapter beyond the algorithms is to understand that there are tradeoffs between having a larger competitor class and a more meaningful definition of the regret that this entails. This is very similar to the tradeoff involved in considering algorithms tuned for a specific environment class (for example, Bernoulli bandits as opposed to bandits with subgaussian noise). Indeed, similarly to what happens when a smaller competitor class is chosen, a more restricted environment class usually allows faster learning, but tuning to a more restricted class runs the risk of losing on performance when the environment that the bandit algorithm runs on does not belong to the restricted class. (The lack of proved guarantees should not be mistaken for the lack of guarantees!)

2 The Exp4 algorithm serves as a tremendous building block for other bandit problems by defining your own experts. The best example of this is the application of Exp4 to nonstationary bandits that we explore in Chapter 31. Here, a combinatorially large set of experts is considered, and yet a fast implementation of Exp4 can be demonstrated to exist. That this is possible is more the exception than the rule. In the lack of such an efficient implementation, Exp4 can still be useful when working with a combinatorially large set of experts just to demonstrate an upper bound on the regret (for an example see Exercise 18.5).

3 The bandits with expert advice framework is clearly more general than contextual bandits. With the terminology of the bandits with expert advice framework, the contextual bandit problem arises when the experts are given by static $\mathcal{C} \to [K]$ maps.

4 A significant challenge is that a naive implementation of Exp4 has running time $O(MK)$ per round, which can be enormous if either M or K is large. In general there is no solution to this problem, but in some cases the computation can be reduced significantly. One situation where this is possible is when the learner has access to an **optimization oracle** that for any context/reward sequence that returns the expert that would collect the most reward in this sequence (this is equivalent to solving the offline problem Eq. (18.8)). In Chapter 30 we show how to use an offline optimization oracle to learn efficiently in combinatorial bandit problems. The idea is to solve a randomly perturbed optimization problem and then show that the randomness in the outputs provides sufficient exploration.

5 In the stochastic contextual bandit problem it is assumed that the context and reward vector form a sequence of independent and identically distributed random variables. Let Φ be a set of $\mathcal{C} \to [K]$ maps and suppose the learner has access to an optimization oracle capable of finding

$$\operatorname{argmax}_{\phi \in \Phi} \sum_{s=1}^{t} x_{s\phi(c_s)}$$

for any sequence of reward vectors $x_1, \dots, x_t$ and contexts $c_1, \dots, c_t$. Under these circumstances there exists a polynomial-time algorithm for which the regret is essentially as the bound in Theorem 18.1.

With access to such an oracle, for stochastic contextual bandit problems there exists a polynomial-time algorithm for which the regret is essentially the same as that stated in Theorem 18.1 [Agarwal et al., 2014]. The algorithm computes importance-weighted estimates of the rewards in each round. These are used to estimate the regret of all the experts. Based on this, a distribution over the experts (with a small support) is computed by solving a feasibility problem: the distribution is constrained so that the importance weights will not be too large, while the regret estimates averaged over the chosen distribution will stay small. To reduce the computation cost, this distribution is updated periodically with the length of the interval between the updates exponentially growing. The significance of this result is that it reduces contextual bandits to (cost-sensitive) empirical risk-minimization (ERM), which means that any advance in solving cost-sensitive ERM problems automatically translates to bandits.

6 The development of efficient algorithms for ERM is a major topic in supervised learning. Note that ERM can be NP-hard even in simple cases like linear classification [Shalev-Shwartz and Ben-David, 2009, §8.7].

7 As noted earlier, the bound on the regret stated in Theorem 18.2 is data-dependent. Thinking of an instance of an adversarial bandit prediction with expert advice problem as the joint choice of the rewards $(x_1, x_2, \dots)$ and the expert predictions $(E^{(1)}, E^{(2)}, \dots)$ we may also call the bound instance-dependent. These two expressions are in fact synonyms of each other, but the stochastic bandit literature mostly uses instance-dependent, while the adversarial online learning literature mostly uses the term data-dependent. In any case, as explained earlier, when a data, or instance-dependent bound is tight enough to imply the worst-case optimal bounds, they are preferred as they give us more information about the algorithm, or, when paired with a matching or nearly matching lower bound, about the problem class.

8 There are many points we have not developed in detail. One is high probability bounds, which we saw in Chapter 12 and can also be derived here. We also have not mentioned lower bounds. The degree to which the bounds are tight depends on whether or not there is additional structure in the experts. In later

chapters we will see examples where the results are essentially tight, but there are also cases where they are not.

18.6 Bibliographic remarks

For a good account on the history of contextual bandits see the article by Tewari and Murphy [2017]. The Exp4 algorithm was introduced by Auer et al. [2002b] and Theorem 18.1 essentially matches Theorem 7.1 of this paper (with a slightly better constant). McMahan and Streeter [2009] noticed that neither the number of experts nor the size of the action set are what really matters for the regret, but rather the extent to which the experts tend to agree. McMahan and Streeter [2009] also introduced the idea of solving a linear program to find an exploration policy that computes a distribution over the actions such that for any action i and round t the computed probability of i is lower bound by the maximum of a constant multiple of $P_t(i)$. This is meant to ensure sufficient exploration while staying close to the output of the exponential weights distribution. The idea of explicitly optimizing a probability distribution with these objectives in mind is at the heart of several works [Agarwal et al., 2014, for example]. While Theorem 18.2 is inspired by this work, the result appears to be new and goes beyond the work of McMahan and Streeter [2009] because it shows that all one needs is to adapt the learning rate based on the degree of agreement amongst the experts. Neu [2015a] proves high probability bounds for Exp4-IX. You can follow in his footsteps by solving Exercise 18.3. Another way to get high probability bounds is to generalize Exp3.P, which was done by Beygelzimer et al. [2011]. As we mentioned in Item 5, there exist efficient algorithms for stochastic contextual bandit problems when a suitable optimization oracle is available [Agarwal et al., 2014]. An earlier attempt to address the problem of reducing contextual bandits to cost-sensitive ERM is by Dudik et al. [2011]. The adversarial case of static experts is considered by Syrgkanis et al. [2016] who prove suboptimal (worse than $\sqrt{n}$) regret bounds under various conditions for follow the perturbed leader for the transductive setting when the contexts are available at the start. The case when the contexts are independent and identically distributed, but the reward is adversarial has been studied by Lazaric and Munos [2009] for the finite expert case, while Rakhlin and Sridharan [2016] considered the case when an ERM oracle is available. The paper of Rakhlin and Sridharan [2016] also considers the more realistic case when only an approximation oracle is available for the ERM problem. What is notable about this work is they demonstrate regret bounds with a moderate blow-up, but without changing the definition. Kakade et al. [2008] consider contextual bandit problems with adversarial context-loss sequences, where all but one action suffers a loss of one in every round. This can also be seen as an instance of **multiclass classification with bandit feedback** where labels to be predicted are identified with actions and the only feedback received is whether the label predicted was correct, with the goal of making as few mistakes as possible. Since minimizing the regret is in general hard in this non-convex setting, just like most of the machine learning literature on classification, Kakade et al. [2008] provide results in the form of mistake bounds for linear classifiers where the baseline is not the number of mistakes of the best linear classifier, but is a convex upper bound on it. The recent book by Shalev-Shwartz and Ben-David [2009] lists some hardness results for ERM. For a more comprehensive treatment, the reader can consult the book by Kearns and Vazirani [1994].

18.7 Exercises

18.1 Let $\mathcal{C}$ be a finite context set and let $c_1, \dots, c_n \in \mathcal{C}$ be an arbitrary sequence of contexts.

(a) Show that $\sum_{c \in \mathcal{C}} \sqrt{\sum_{t=1}^n \mathbb{I}\{c_t = c\}} \leq \sqrt{n|\mathcal{C}|}$.

(b) Assume that n is an integer multiple of $|\mathcal{C}|$. Show that the choice that maximizes the right-hand side of the previous inequality is the one when each context occurs $n/|\mathcal{C}|$ times.

18.2 Prove Lemma 18.1.

18.3 In this exercise you will prove an analogue of Theorem 12.1 for Exp4-IX. In the contextual setting the random regret is

$$\hat{R}_n = \max_{m \in [M]} \sum_{t=1}^n \left(E_m^{(t)} x_t - X_t\right) .$$

Design an algorithm accepting parameter $\delta \in (0,1)$ such that

$$\mathbb{P}\left(\hat{R}_n \geq C\left(\sqrt{nK \log(K)} + \sqrt{\frac{nK}{\log(K)}} \log\left(\frac{1}{\delta}\right)\right)\right) \leq \delta .$$

18.4 Prove Theorem 18.2.

18.5 Let $x_1, \dots, x_n$ be a sequence of reward vectors chosen in advance by an adversary with $x_t \in [0,1]^K$. Furthermore, let $o_1, \dots, o_n$ be a sequence of observations, also chosen in advance by an adversary with $o_t \in [O]$ for some fixed $O \in \mathbb{N}^+$. Then let $\mathcal{H}$ be the set of functions $\phi : [O]^m \to [K]$ where $m \in \mathbb{N}^+$. In each round the learner observes o_t should choose an action A_t based on $o_1, A_1, X_1, \dots, o_{t-1}, A_{t-1}, X_{t-1}, o_t$ and the regret is

$$R_n = \min_{\phi \in \mathcal{H}} \sum_{t=1}^n x_{tA_t} - x_{t\phi(o_t, o_{t-1}, \dots, o_{t-m})} ,$$

where $o_t = 1$ for $t \leq 0$. This means the learner is competing with the best predictor in hindsight that uses only the last m observations. Prove there exists an algorithm such that

$$\mathbb{E}[R_n] \leq \sqrt{2nmK\log(O)}\,.$$

18.6 In this problem we consider non-oblivious experts. Consider the following modified regret definition:

$$R'_n = \max_{m \in [M]} \mathbb{E}\left[\sum_{t=1}^n E_m^{(t)} x_t - \sum_{t=1}^n X_t\right]\,.$$

Show that:

(a) $R'_n \leq R_n$ regardless of whether the experts are oblivious or not.
(b) Theorem 18.1 remains valid for non-oblivious experts if in Eq. (18.7) we replace R_n with R'_n. In particular, explain how to modify the proof.
(c) Research question: Give a non-trivial bound on R_n.

18.7 Prove Eq. (18.13).

18.8 [The epoch-greedy algorithm, [Langford and Zhang, 2008]] Consider a stochastic contextual bandit environment, where the context-reward pairs (C_t, X_t) form an i.i.d. sequence, with $C_t \in \mathcal{C}$ and $X_t \in [0,1]^K$. Let $\Phi \subset \{\phi : \phi : \mathcal{C} \to [K]\}$ be a set of static experts and assume that we have access to an oracle $\mathcal{O}(x, c)$ that can compute $\operatorname{argmax}_{\phi \in \Phi} \sum_{s=1}^t x_{t,\phi(c_t)}$ for any $x = (x_s)_s$, $c = (c_s)_s$ sequences of reward-vectors and contexts ($x_s \in \mathbb{R}^K$, $c_s \in \mathcal{C}$).

The **epoch-greedy algorithm** works in phases of length $1 < \tau_1 < \tau_2 < \dots$ of increasing length. In the first round of phase $m = 1, 2, \dots$, the algorithm receives context $\tilde{C}_m$ and then performs an exploration step: An action $\tilde{A}_m \in [K]$ is chosen uniformly at random. Let $\tilde{X}_m$ denote the reward received in response. Next, the algorithm constructs the reward estimates $\hat{X}_{m,k} = \frac{1}{K}\mathbb{I}\left\{\tilde{A}_m = k\right\}\tilde{X}_m$ and finds the expert ϕ_m whose usage so far would have incurred the most total reward: $\phi_m = \operatorname{argmax}_{\phi \in \Phi} \sum_{p=1}^m \hat{X}_{p,\phi(\tilde{C}_p)}$. In the remaining $\tau_m - 1$ rounds of phase m, the advice of ϕ_m is followed: Upon receiving context C_t in round t (during this phase), action $A_t = \phi_m(C_t)$ is used.

(a) Let Φ be finite. Show that with an appropriate choice of $(\tau_m)_m$, the expected regret R_n of epoch-greedy after n steps is $R_n \leq O(n^{2/3}|\Phi|^{1/3})$.
(b) Extension to VC-dimension!
(c) Can the result be extended to the case when the context sequence $(c_t)_t$ is an arbitrary fixed sequence, but $X_t \sim P_{c_t}$ for some family $(P_c)_c$ of distributions?