Online-Lernen

Autor und Herausgeber

Alexander Jung*, Department of Computer Science, Aalto University, Espoo, Finland (ORCID 0000-0001-7538-0990)
Editors: Konstantina Olioumtsevits (ORCID 0000-0003-2697-8659), Ekkehard Schnoor (ORCID 0000-0001-8252-1499), Aalto University, Espoo, Finland
*Corresponding author: alex.jung@aalto.fi

Zusammenfassung

Online-Lernen bezieht sich auf machine learning (ML)-Methoden, die die Modellparameter kontinuierlich aktualisieren, wenn im Laufe der Zeit neue Datenpunkte eintreffen. Dies steht im Gegensatz zu Offline-Methoden wie der lineare Regression, die die empirische Risikominimierung (ERM) einmal über einen festen Trainingsdatensatz lösen. Die Aktualisierungen basieren auf einer Form von Feedback: Beispiele sind eine gesamte Verlustfunktion, die durch den neuen Datenpunkt bestimmt wird, oder ein Belohnung-Signal. Der Online gradient descent (online GD) verwendet eine gesamte Verlustfunktion als Feedback; Bandit- und bestärkendes Lernen (RL)-Methoden ersetzen den nicht verfügbaren Gradient durch eine Schätzung aus dem Feedback. Die Gütezahl ist der Regret: der akkumulierte Verlust abzüglich des Verlusts fester Modellparameter, die im Nachhinein gewählt wurden. Eine Methode ist erfolgreich, wenn ihr Regret pro Runde mit zunehmender Rundenzahl verschwindet.

Definition

Eine Wetterstation meldet stündlich einen Temperaturwert, und eine Vorhersage für die nächste Stunde muss abgegeben werden, bevor dieser Wert eintrifft. Eine Offline-Methode wie die lineare Regression sammelt zunächst Messwerte in einem Trainingsdatensatz und trainiert einmal ein Modell darauf. Sie prognostiziert dann mit den gelernten Modellparameter, die von da an fest bleiben: Während sie sammelt, gibt sie keine Vorhersage aus, und nachdem sie trainiert hat, lernt sie nicht mehr. Eine Online-Lernmethode passt stattdessen ihre Modellparameter jedes Mal an, wenn ein neuer Messwert eintrifft; Abb. 1 zeigt beides anhand von Messwerten von drei Tagen.

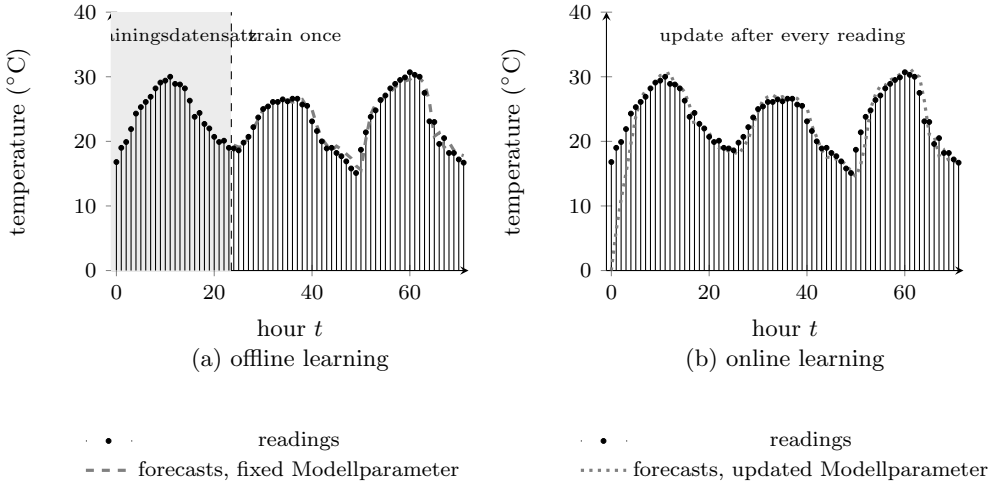


Abbildung 1: Offline- und Online-Lernen eines Temperaturvorhersagers auf drei Tagen stündlicher Messungen in Krems an der Donau (Striche), jeweils Vorhersage jeder Stunde aus den Messungen der drei vorherigen Stunden. (a) Eine Offline-Methode sammelt die Messungen des ersten Tages zu einem Trainingsdatensatz, trainiert einmal einen lineare Regression-Vorhersager darauf und sagt die nächsten zwei Tage mit den gelernten Modellparameter $\widehat{\mathbf{w}}$ voraus; während des ersten Tages gibt sie keine Vorhersage ab. (b) Eine Online-Methode startet mit $\mathbf{w}^{(1)} = \mathbf{0}$ zur Stunde 0, gibt zu jeder Stunde eine Vorhersage mit ihren aktuellen Modellparameter $\mathbf{w}^{(t)}$ ab und aktualisiert diese, sobald die Messung eintrifft; ihre ersten Vorhersagen liegen weit daneben, und innerhalb des ersten Tages folgt sie den Messungen. Daten erzeugt von `pythondemos/onlinelearning.py`

Online-Lernen verläuft in Runden: In jeder Runde legt sich eine machine learning (ML)-Methode auf Modellparameter fest, empfängt ein Feedback-Signal von ihrer environment und aktualisiert (Cesa-Bianchi and Lugosi, 2006; Hazan, 2022, Sect. 1.1; Shalev-Shwartz, 2012). Die Aktualisierungen basieren auf einer Form von Feedback. Für die Wetterstation ist das Feedback der neue Datenpunkt selbst. Jeder neue Temperaturwert ermöglicht die Berechnung des squared error loss gegenüber den aktuellen Modellparameter, und der Verlust kann für jede mögliche Wahl der Modellparameter bewertet werden. Das Feedback ist daher eine gesamte Verlustfunktion $\mathbf{w} \mapsto L\left(\mathbf{z}^{(t)}, \mathbf{w}\right)$ (Hazan, 2022, Sect. 1.1). Der Online gradient descent (online GD) ist die Online-Lernmethode für Feedback durch eine gesamte Verlustfunktion: Er aktualisiert die Modellparameter durch einen Gradientenschritt auf dieser Verlustfunktion (Zinkevich, 2003).

Einige Online-Lernanwendungen liefern nur einen einzelnen Verlust- oder Belohnung-Wert als Feedback, im Gegensatz zu vollständigen Informationen als Bandit-Feedback bezeichnet (Abb. 2). Zum Beispiel beobachtet eine Klinik, die

eine von mehreren Behandlungen verabreicht, nur das Ergebnis dieser Behandlung, was ein multiarmed bandit (MAB) ist (Lattimore and Szepesvári, 2020, Ch. 1). Ein Thermostat, dessen Heizentscheidung die Temperatur ändert, der er in der nächsten Stunde ausgesetzt ist, erhält eine Belohnung zusammen mit einem neuen Zustand, was bestärkendes Lernen (RL) ist (Sutton and Barto, 2018, Ch. 3).

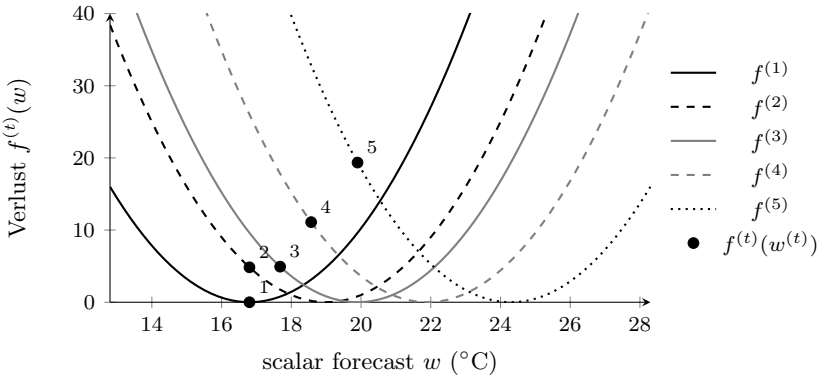


Abbildung 2: Die Verlustfunktionen von fünf aufeinanderfolgenden Stunden des Temperaturstroms in einer Ebene ($f^{(1)}$ schwarze durchgezogene Linie, $f^{(2)}$ schwarze gestrichelte Linie, $f^{(3)}$ graue durchgezogene Linie, $f^{(4)}$ graue gestrichelte Linie und $f^{(5)}$ schwarze gepunktete Kurve), für den skalaren Vorhersager w und die squared error loss $f^{(t)}(w) = (y^{(t)} - w)^2$, die beim Messwert $y^{(t)}$ der Stunde t minimal ist. Bei vollständiger Information wird die gesamte Kurve $f^{(t)}$ zur Stunde t offenbart. Bei Bandit- Feedback wird nur ihr Wert bei der festgelegten Vorhersage $w^{(t)}$ bekannt (Punkte, mit t beschriftet); hier sind die Vorhersagen die Iterierten von online GD, jeweils um einen festen Bruchteil der Distanz zum neuesten Messwert verschoben. Daten erzeugt von `pythondemos/onlinelearning.py`

Man betrachte Modellparameter $\mathbf{w}$ aus einer konvexen closed set $\mathcal{W} \subseteq \mathbb{R}^d$, dem parameter space, und Runden zu aufeinanderfolgenden Zeitpunkten $t = 1, 2, \ldots$. Zum Zeitpunkt t legt die environment eine Funktion $f^{(t)} : \mathcal{W} \to \mathbb{R}$ fest, und die aktuellen Modellparameter $\mathbf{w}^{(t)}$ verursachen den Verlust $f^{(t)}(\mathbf{w}^{(t)})$. Erst dann wird ein Feedback-Signal $r^{(t)}$ über $f^{(t)}$ offenbart. Das Feedback-Signal wird als $r^{(t)}$ geschrieben, in der Belohnung-Notation des RL, obwohl es bei vollständigen Informationen ein Datenpunkt und bei Bandit-Feedback ein Verlust-Wert ist. Drei Arten von Feedback treten auf.

Bei vollständigen Informationen wird der Datenpunkt $\mathbf{z}^{(t)}$ offenbart, und er bestimmt $f^{(t)}(\mathbf{w}) = L\left(\mathbf{z}^{(t)}, \mathbf{w}\right)$ für jedes $\mathbf{w} \in \mathcal{W}$. Im Temperaturstrom hat der Datenpunkt $\mathbf{z}^{(t)} = (\mathbf{x}^{(t)}, y^{(t)})$ als Merkmale, gesammelt im Merkmalsvektor $\mathbf{x}^{(t)}$, die Messwerte der vorherigen Stunden und als Label den Messwert der Stunde t. Seine Verlustfunktion $f^{(t)}$ ist der squared error loss der Vorhersage $h^{(\mathbf{w})}(\mathbf{x}^{(t)})$.

Bei Bandit-Feedback wird nur der Wert $f^{(t)}(\mathbf{w}^{(t)})$ offenbart. In einem MAB mit A Aktionen bilden die Modellparameter $\mathbf{w}^{(t)} \in \Delta^{A-1}$ die probability mass function (pmf), aus der die Aktion $a^{(t)}$ gezogen wird. Die environment legt einen Verlust-Vektor $\boldsymbol{\ell}^{(t)} \in [0,1]^A$ fest, dessen Eintrag $\ell_a^{(t)}$ der Verlust der Aktion a in Runde t ist, sodass der erwartete Verlust der Ziehung $f^{(t)}(\mathbf{w}) = \langle \boldsymbol{\ell}^{(t)}, \mathbf{w} \rangle$ beträgt. Die Klinik beobachtet den einzelnen Eintrag $\ell_{a^{(t)}}^{(t)}$, den Verlust der gezogenen Aktion, und keinen anderen Eintrag.

Bei RL-Feedback werden die unmittelbare Belohnung $r(s^{(t)}, a^{(t)})$ der ausgeführten Aktion und der nächste Zustand $s^{(t+1)}$ eines Markov decision process (MDP) offenbart. Der Verlust $f^{(t)}(\mathbf{w})$ ist die negative erwartete Belohnung der Policy (bestärkendes Lernen) mit Modellparameter $\mathbf{w}$ im Zustand $s^{(t)}$. Dieser Zustand hängt von den Aktionen der früheren Runden ab, sodass die Verluste der environment von der eigenen Vergangenheit der Methode abhängen.

Die Methode aktualisiert dann die Modellparameter unter Verwendung des Feedbacks,

$$\mathbf{w}^{(t+1)} := \mathcal{F}^{(t)}\big(\mathbf{w}^{(t)}, r^{(t)}\big), \tag{1}$$

mit einem für die Methode spezifischen Update-Operator $\mathcal{F}^{(t)}$, dem Gegenstück zum Operator $\mathcal{F}$ einer fixed-point iteration (Abb. 3). Der Zeitindex ist auch der Iterationsindex: eine Aktualisierung für jeden neu eintreffenden Datenpunkt. Dies steht im Gegensatz zu batch learning, das die empirische Risikominimierung (ERM) einmal über einen festen Trainingsdatensatz löst und dann die gelernte Hypothese verwendet.

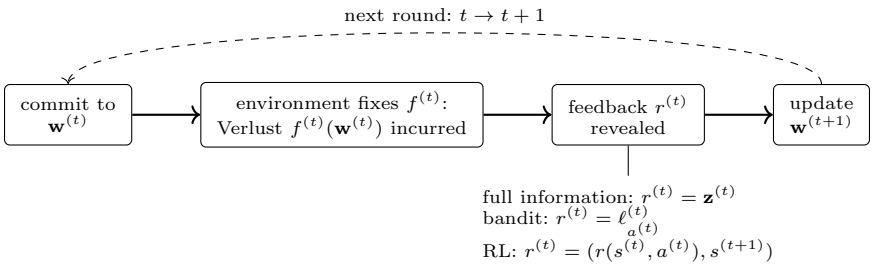


Abbildung 3: Eine Runde Online-Lernen. Die Methode legt sich auf die Modellparameter $\mathbf{w}^{(t)}$ fest, bevor die environment die Verlust $f^{(t)}$ der Runde bestimmt, erleidet $f^{(t)}(\mathbf{w}^{(t)})$, erhält ein Feedback- Signal $r^{(t)}$ und aktualisiert auf $\mathbf{w}^{(t+1)}$ für die nächste Runde. Das Feedback ist der Datenpunkt selbst (vollständige Information, wie bei online GD), nur die Verlust der gezogenen Aktion (Bandit) oder eine Belohnung und der nächste Zustand (RL)

Die Aktualisierung (1) zum Zeitpunkt t wendet einen Operator an, der vom Feedback-Signal $r^{(t)}$ abhängt. Im Gegensatz dazu wendet der Gradientenabstieg in jedem Zeitschritt einen festen Gradientenschritt-Operator an. Eine Online-Lernmethode ist daher keine fixed-point iteration eines festen Operators. Stattdessen verkettet sie Operatoren, die über die Zeitschritte variieren. Anders als

der stochastischer Gradientenabstieg (SGD), der einen Datenpunkt pro Schritt verwendet, um den Gradient einer festen Zielfunktion zu approximieren, sieht sich eine Online-Lernmethode in jeder Runde einer anderen Verlustfunktion gegenüber.

Bei vollständigen Informationen verwendet der Online GD den Gradient $\mathbf{g}^{(t)} := \nabla f^{(t)}(\mathbf{w}^{(t)})$ in einem projizierten Gradientenschritt. Der Online mirror descent (online MD) ersetzt die euklidischer Abstand der Projektion durch eine Bregman divergence, z.B. die Kullback–Leibler divergence (KL divergence) auf dem probability simplex, was das Hedge-Update der pmf über die Aktionen ergibt (Hazan, 2022, Sect. 5.4). Follow-The-Regularized-Leader (FTRL) legt sich in jeder Runde auf einen Minimierer des akkumulierten Verlust aller bisherigen Runden plus eines Regularisierungsterms fest (Hazan, 2022, Sect. 5.2).

Bei Bandit-Feedback ist der Gradient von $f^{(t)}$ nicht verfügbar, da nur ein Eintrag von $\boldsymbol{\ell}^{(t)}$ beobachtet wird. Exp3 bildet eine unverzerrte Schätzung von $\boldsymbol{\ell}^{(t)}$, indem es den beobachteten Eintrag durch die Wahrscheinlichkeit der gezogenen Aktion teilt, und wendet das online MD-Update auf die Schätzung an (Auer et al., 2002, Sect. 3).

Bei RL-Feedback ändert Q-learning einen Eintrag einer Tabelle von Aktion-Werten, den Eintrag des besuchten Zustands und der ausgeführten Aktion. Der Eintrag verschiebt sich um einen festen Bruchteil der Differenz zwischen seinem Ziel, der erhaltenen Belohnung plus dem diskontierten größten Wert des nächsten Zustands, und seinem aktuellen Wert. Die Aktualisierung ist ein stochastischer Schritt des Bellman operators, eine Kontraktion mit dem Faktor γ in der supremum (or least upper bound)-Norm: Die Wahl des größten Wertes über die Aktionen kann die größte Differenz zwischen zwei Wertschätzungen nicht erhöhen, und der Diskontierungsfaktor skaliert diese Differenz um γ. Q-learning hat, im Gegensatz zum Online GD, eine Fixpunkt-Interpretation (Sutton and Barto, 2018, Sect. 6.5). Policy gradient-Methoden schätzen stattdessen den Gradient der erwarteten Belohnung aus den Belohnungen, die entlang der besuchten Zustände gesammelt wurden (Sutton and Barto, 2018, Ch. 13).

Die Gütezahl einer Online-Lernmethode ist nicht die Abnahme einer Zielfunktion entlang der Iterationen, die den Fortschritt des Gradientenabstieg misst: Online-Lernen hat keine feste Zielfunktion. Die Verlustfunktion $f^{(t)}$ der Runde t wird einmal um $\mathbf{w}^{(t)}$ herum ausgewertet, und die nächste Runde bringt eine andere Verlustfunktion. Stattdessen wird eine Online-Lernmethode durch ihren Regret bewertet.

Der Regret ist der übermäßige akkumulierte Verlust im Vergleich zu dem einer festen Wahl $\mathbf{w} \in \mathcal{W}$ der Modellparameter. Diese feste Wahl $\mathbf{w}$ wird für alle Runden $t = 1, \ldots, T$ beibehalten und kann gewählt werden, nachdem alle Verluste bekannt sind. Nach T Runden ist der Regret relativ zu $\mathbf{w} \in \mathcal{W}$ definiert als

$$\text{Regret}_T(\mathbf{w}) := \sum_{t=1}^{T} f^{(t)}(\mathbf{w}^{(t)}) - \sum_{t=1}^{T} f^{(t)}(\mathbf{w}).$$

In einem MAB ist das relevante feste $\mathbf{w}$ die auf eine Aktion konzentrierte pmf. Der Regret wird dann gegen die im Nachhinein beste einzelne Aktion gemessen;

im RL gegen die beste feste Policy (bestärkendes Lernen) (Jaksch et al., 2010, Sekt. 2).

Es wird keine Annahme darüber gemacht, wie die Verluste erzeugt werden: Der Regret ist für jede Folge $f^{(1)}, f^{(2)}, \ldots$ definiert, auch für eine, die von einem Gegner gewählt wird, der $\mathbf{w}^{(t)}$ gesehen hat. Eine Methode des Online-Lernens ist erfolgreich, wenn ihr Regret pro Runde, $\text{Regret}_T(\mathbf{w})/T$, für jedes $\mathbf{w} \in \mathcal{W}$ gegen null geht, wenn T wächst (Shalev-Shwartz, 2012, Sekt. 1.1).

Regret-Schranken. Die Geschwindigkeit, mit der der Regret pro Runde gegen null geht, hängt vom Feedback ab. Bei vollständiger Information und konvexen Verluste, deren Gradienten eine Norm von höchstens G auf einem parameter space mit Durchmesser D haben, gilt für online GD mit der Lernrate $\eta^{(t)} = D/(G\sqrt{t})$ die Schranke $\text{Regret}_T(\mathbf{w}) \leq \frac{3}{2} DG\sqrt{T}$ (Hazan, 2022, Thm. 3.1). Ist die Rundenzahl T im Voraus bekannt, so gilt für die konstante Lernrate $\eta = D/(G\sqrt{T})$ die Schranke $\text{Regret}_T(\mathbf{w}) \leq DG\sqrt{T}$ nach demselben Argument. Kein online algorithm kann einen Regret unterhalb eines konstanten Vielfachen von $DG\sqrt{T}$ für jede Folge garantieren: Zu jedem Algorithmus gibt es eine Folge von Verluste, auf der sein Regret mindestens so groß ist (Hazan, 2022, Thm. 3.2).

Bei Bandit-Feedback über A Aktionen und Verluste in $[0, 1]$ ist der erwartete Regret von Exp3 höchstens $2\sqrt{(e-1)TA\ln A}$ (Auer et al., 2002, Kor. 3.2): Das Beobachten eines Eintrags von $\boldsymbol{\ell}^{(t)}$ statt aller kostet den Faktor $\sqrt{A}$. Bei RL- Feedback in einem MDP mit S Zustände und A Aktionen hat die UCRL2-Methode mit Wahrscheinlichkeit mindestens $1 - \delta$ für jedes $\delta \in (0, 1)$ und jedes $T > 1$ einen Regret von höchstens einem konstanten Vielfachen von $S\sqrt{AT\ln(T/\delta)}$, wobei die Konstante davon abhängt, wie viele Schritte es braucht, um von jedem Zustand zu jedem anderen zu gelangen (Jaksch et al., 2010, Thm. 2). In jedem Fall geht der Regret pro Runde gegen null, wenn T wächst.

Abb. 4 zeigt den Regret pro Runde von online GD und von Follow-the-Leader, der FTRL- Methode ohne ihren Regularisierungsterm, die in jeder Runde einen Minimierer der bisher akkumulierten Verlust wählt, auf einem Jahr stündlicher Temperaturmessungen. Für lineare Regression mit der squared error loss wird dieser Minimierer durch das rekursive Update der kleinsten Quadrate berechnet, das die Lösung der kleinsten Quadrate von Zeit $t-1$ zu Zeit t in geschlossener Form mit der Sherman–Morrison-Formel für die Inverse einer Matrix nach einer Rang-eins-Änderung überträgt (Golub and Van Loan, 2013, Sekt. 2.1.4). Online GD wird für jede gezeigte Rundenzahl T neu gestartet, mit der konstanten Lernrate $\eta = D/(G\sqrt{T})$ auf einem parameter space mit Durchmesser $D = 4$ und mit G als größter aufgetretener Gradient-Norm, sodass die Schranke $DG/\sqrt{T}$ auf den Regret pro Runde für jedes gezeigte T gilt.

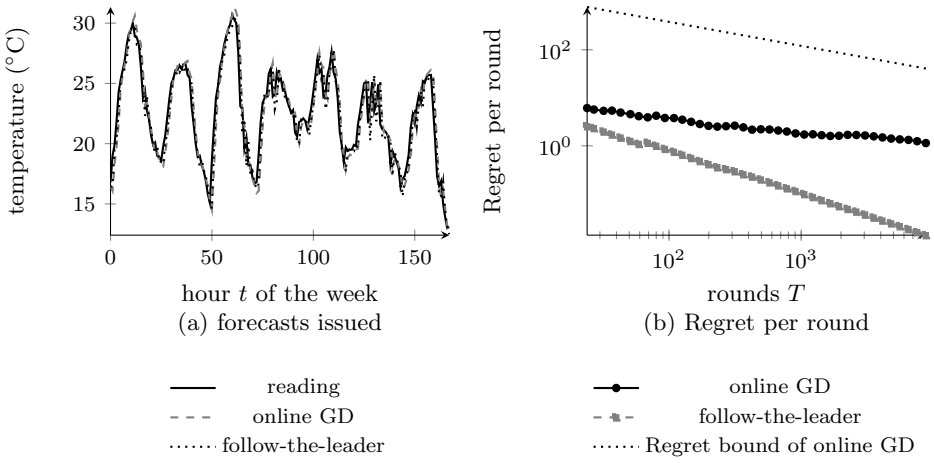


Abbildung 4: Online-Lernen eines Temperaturvorhersagers für Krems an der Donau im Jahr 2024. Jede neue Temperaturmessung liefert einen neuen Datenpunkt, mit den Messungen der drei vorherigen Stunden als Merkmale und der squared error loss. (a) Eine Woche Messungen und die Vorhersagen, die von online GD und von Follow-the-Leader vor jeder Messung abgegeben wurden. (b) Der Regret pro Runde beider Methoden nach T Runden, relativ zu den im Nachhinein besten Modellparameter über dieselben Runden, und die Schranke $DG/\sqrt{T}$ von online GD, ausgeführt mit der Lernrate $D/(G\sqrt{T})$ für jedes T. Daten erzeugt von `pythondemos/onlinelearning.py`

Abb. 5 verwandelt denselben Strom in ein MAB. Die Aktionen sind fünf feste Vorhersager, jeweils eine Regel, die kein Training benötigt. Sie sagen den vorherigen Wert voraus, den Wert derselben Stunde einen Tag zuvor, den Mittelwert der drei vorherigen Werte, die lineare Extrapolation der beiden vorherigen Werte und den Mittelwert aller bisherigen Werte des Jahres. Die Verlust einer Stunde ist der skalierte absolute Fehler des verwendeten Vorhersagers, und der Regret wird gegen den im Nachhinein besten einzelnen Vorhersager gemessen.

Bei vollständiger Information werden nach jeder Stunde alle fünf Fehler beobachtet und Hedge aktualisiert die pmf; bei Bandit- Feedback wird nur der Fehler des verwendeten Vorhersagers beobachtet und Exp3 aktualisiert die pmf aus seiner Schätzung des Verlust-Vektor.

Der Regret von Hedge nach T Runden ist höchstens $\sqrt{2T \ln A}$ für Verluste in $[0, 1]$, eine Abschwächung der Schranke $\ln A/\eta + \eta T/8$ aus Cesa-Bianchi and Lugosi (2006), Thm. 2.2 bei der Lernrate $\eta = \sqrt{2 \ln A/T}$; für Exp3 gilt die angegebene Schranke auf den erwarteten Regret bei Bandit-Feedback mit $A = 5$. Beide Regrets pro Runde gehen gegen null, und der Regret von Exp3 bleibt größer als der von Hedge, was den durch Bandit-Feedback verursachten Anstieg des Regret zeigt.

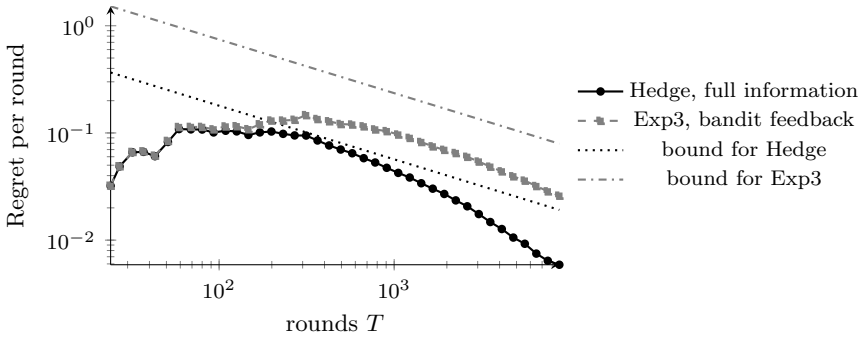


Abbildung 5: Der Temperaturstrom als MAB mit $A = 5$ festen Vorhersagern als Aktionen. Der Regret pro Runde nach T Runden gegenüber dem im Nachhinein besten einzelnen Vorhersager, für Hedge mit vollständiger Information (durchgezogene Kurve, ausgefüllte Kreise) und für Exp3 mit Bandit-Feedback (gestrichelte Kurve, ausgefüllte Quadrate), zusammen mit den Schranken $\sqrt{2 \ln A/T}$ von Hedge (gepunktete Kurve) und $2\sqrt{(e-1)A \ln A/T}$ von Exp3 (strichpunktierte Kurve). Daten erzeugt von `pythondemos/onlinelearning.py`

Siehe auch: online GD, online algorithm, Regret, Hedge, Exp3, online MD, mirror descent, FTRL, SGD, Gradientenabstieg, Gradientenschritt, ERM, Risiko, MAB, RL, MDP, Q-learning, batch learning, fixed-point iteration, lineare Regression.

Querverweise

Literatur

1. Auer et al. (2002). The Nonstochastic Multiarmed Bandit Problem. SIAM J. Comput., 32(1), pp. 48–77.

2. Golub and Van Loan (2013). Matrix Computations. 4th ed., The Johns Hopkins Univ. Press, Baltimore, MD, USA.

3. Hazan (2022). Introduction to Online Convex Optimization. 2nd ed., MIT Press, Cambridge, MA, USA.

4. Jaksch et al. (2010). Near-optimal Regret Bounds for Reinforcement Learning. Journal of Machine Learning Research, 11, pp. 1563–1600.

5. Lattimore and Szepesvári (2020). Bandit Algorithms. Cambridge University Press, Cambridge, UK.

6. Cesa-Bianchi and Lugosi (2006). Prediction, Learning, and Games. Cambridge Univ. Press, New York, NY, USA.

7. Shalev-Shwartz (2012). Online Learning and Online Convex Optimization. Foundations and Trends in Machine Learning, 4(2), pp. 107–194.

8. Sutton and Barto (2018). Reinforcement Learning: An Introduction. 2nd ed., MIT Press, Cambridge, MA, USA.

9. Zinkevich (2003). Online Convex Programming and Generalized Infinitesimal Gradient Ascent. In: Proc. 20th Int. Conf. Mach. Learn., pp. 928–935.