Lernrate

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

Eine Lernrate ist ein Hyperparameter einer iterativen machine learning (ML)-Methode, der das Ausmaß einer einzelnen Aktualisierung der aktuellen Hypothese steuert. Ein Paradebeispiel für eine Lernrate ist die Schrittweite des Gradientenabstieg. Eine Lernrate, die zu klein ist, macht wenig Fortschritt innerhalb der verfügbaren Anzahl von Iterationen, und eine, die zu groß ist, kann zu divergenten Iterationen führen. Für eine feste Anzahl von Iterationen begrenzt die Lernrate den effektiven Hypothesenraum einer ML-Methode, was sie zu einem Mittel zur Regularisierung macht. Die Lernrate kann durch den Validierungsfehler der gelernten Hypothese gewählt werden.

Definition

Das Lernen eines lineares Modells mittels Gradientenabstieg auf dem durchschnittlichen squared error loss über einem Trainingsdatensatz, wie in Abb. 1, ist eine iterative machine learning (ML)-Methode zum Lernen einer nützlichen Hypothese $h \in \mathcal{H}$. Eine solche Methode wiederholt ähnliche Aktualisierungsschritte, die die aktuelle Hypothese modifizieren, um eine verbesserte zu erhalten. Eine Lernrate ist ein Parameter einer solchen Methode, der das Ausmaß steuert, in dem die aktuelle Hypothese während einer einzelnen Iteration modifiziert werden kann.

Das Hauptbeispiel für eine Lernrate η ist die Schrittweite des Gradientenabstieg (Boyd and Vandenberghe, 2004, Sect. 9.3),

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \nabla f(\mathbf{w}^{(t)}). \tag{1}$$

Hier ist die Zielfunktion $f(\mathbf{w})$ das empirical risk, das durch eine Hypothese $h^{(\mathbf{w})}$ auf einem Trainingsdatensatz entsteht, und jeder Gradientenschritt bewegt die Modellparameter entgegen ihrem Gradient $\nabla f(\mathbf{w}^{(t)})$ um einen Abstand, der proportional zu η ist. Abb. 1 veranschaulicht einen Gradientenschritt (1) des Gradientenabstieg für ein lineares Modell, ausgehend von derselben Hypothese mit drei Wahlmöglichkeiten der Lernrate. Die Lernraten sind als Vielfache von $1/L$ angegeben, wobei L der größte Eigenwert der Hessian von f ist. Für den squared error loss senkt ein Gradientenschritt f für jede Lernrate unter $2/L$ (Nesterov, 2004, Thm. 2.1.14); die größte Lernrate in Abb. 1 liegt über diesem Schwellenwert, und ihr Schritt erhöht f.

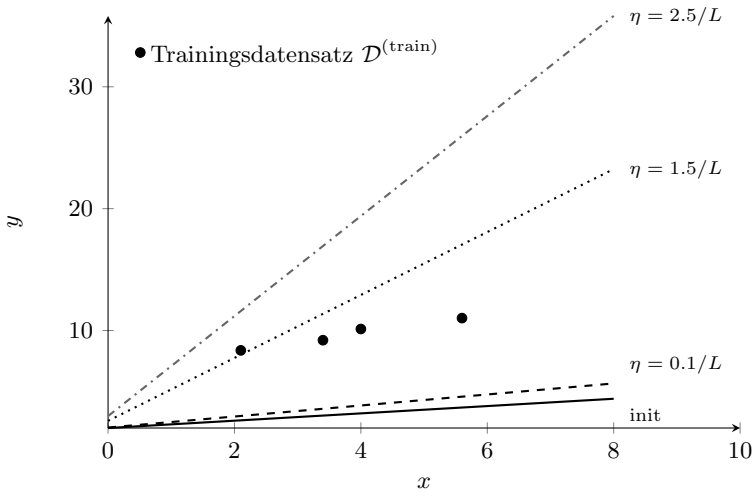


Abbildung 1: Ein Gradientenschritt des Gradientenabstieg auf dem durchschnittlichen squared error loss eines lineares Modells auf einem Trainingsdatensatz. Die durchgezogene Linie ist die Hypothese-Abbildung vor dem Schritt. Die gestrichelte Linie ist die Hypothese-Abbildung nach einem Schritt mit der Lernrate $\eta = 0.1/L$, die gepunktete Linie nach einem Schritt mit $\eta = 1.5/L$, und die grau-strichpunktierte Linie nach einem Schritt mit $\eta = 2.5/L$, wobei L der größte Eigenwert der Hessian der Zielfunktion ist. Daten generiert von `pythondemos/learnrate.py`

Jede Iteration einer ML-Methode sollte große Fortschritte in Richtung einer nützlichen Hypothese machen, daher sollte die Lernrate nicht zu klein sein. Der Schritt mit $\eta = 2.5/L$ in Abb. 1 zeigt das andere Versagen: eine zu große Schrittweite des Gradientenabstieg lässt die Gradientenschritte ein minimum der Zielfunktion überschießen. Abb. 2 veranschaulicht beide Fälle.

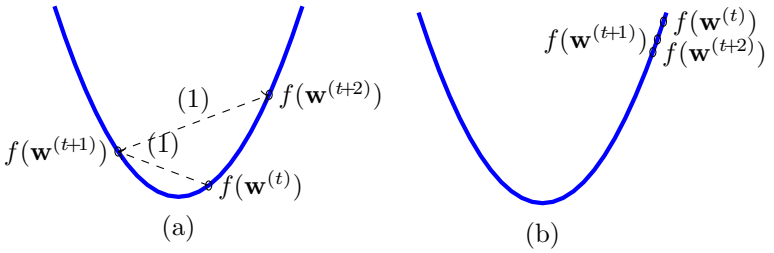


Abbildung 2: Auswirkung der Verwendung einer unzureichenden Schrittweite η im Gradientenschritt (1). (a) Wenn η zu groß ist, können die Gradientenschritte überschießen, sodass sich die Iterierten $\mathbf{w}^{(t)}$ von einem minimum von f entfernen, d.h. $f(\mathbf{w}^{(t+1)}) > f(\mathbf{w}^{(t)})$. (b) Wenn η zu klein ist, machen die Gradientenschritte zu wenig Fortschritt in Richtung eines minimums innerhalb der verfügbaren Anzahl von Iterationen

Betrachten Sie eine iterative ML-Methode, die eine Hypothese aus einem zugrunde liegenden Hypothesenraum $\mathcal{H}$ lernt, indem sie eine feste Anzahl T von Iterationen durchführt. Das Variieren der Lernrate implementiert Regularisierung, indem es den effektiven $\mathcal{H}$ beschneidet. Zum Beispiel bewegt jeder Gradientenschritt oben (1) die Modellparameter um $\eta\|\nabla f(\mathbf{w}^{(t)})\|$. Nach der Dreiecksungleichung gilt:

$$\|\mathbf{w}^{(T)} - \mathbf{w}^{(0)}\| \leq \eta \sum_{t=0}^{T-1} \|\nabla f(\mathbf{w}^{(t)})\|. \tag{2}$$

Zusammen mit T bestimmt die Lernrate somit den Bereich des parameter space, den die Methode erreichen kann, und damit den effektiven Hypothesenraum: die Hypothesen $h^{(\mathbf{w})}$, deren Modellparameter $\mathbf{w}$ in diesem Bereich liegen.

Abb. 3 zeigt die Kugeln um $\mathbf{w}^{(0)}$ mit dem Radius (2) für $T = 10$ Gradientenschritte des Gradientenabstieg, die das empirische Risikominimierung (ERM)-Problem der lineare Regression lösen, und drei Lernraten. Das Reduzieren der Lernrate verkleinert den effektiven Hypothesenraum, der durch die Gradientenschritte erreicht werden kann. Eine zu kleine oder zu große Lernrate kann dann zu underfitting oder Überanpassung führen.

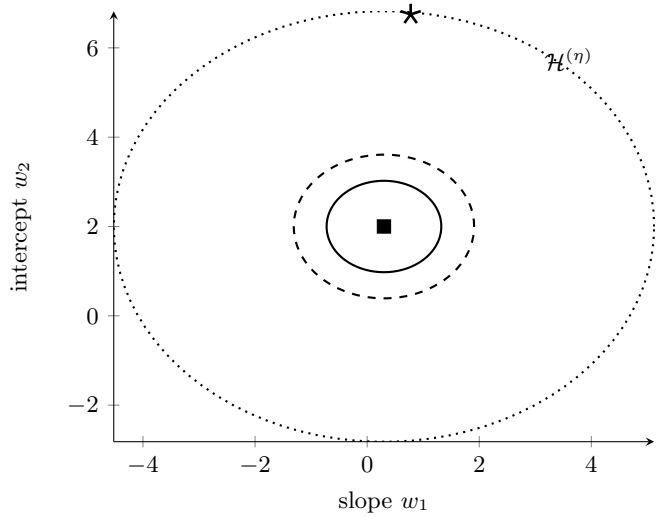


Abbildung 3: Die Modellparameter, die $T = 10$ Gradientenschritte für die lineare Regression vom anfänglichen $\mathbf{w}^{(0)}$ (gefülltes Quadrat) aus erreichen können; jede Kugel ist der effektive Hypothesenraum $\mathcal{H}^{(\eta)}$ ihrer Lernrate. Jede Kurve ist die Kugel um $\mathbf{w}^{(0)}$ mit dem Radius (2), berechnet entlang der Iterationen mit der Lernrate $\eta = 0.1/L$ (durchgezogene Kurve), $0.5/L$ (gestrichelte Kurve) und $1.5/L$ (gepunktete Kurve). Die ERM-Lösung $\widehat{\mathbf{w}}$ (Stern) liegt außerhalb der beiden kleineren Kugeln und innerhalb der größten. Daten generiert von `pythondemos/learnrate.py`

Als Hyperparameter kann die Lernrate durch validation gewählt werden: Die mit verschiedenen Lernraten gelernten Hypothesen werden anhand ihres Validierungsfehler auf einem Validierungsdatensatz verglichen.

Synonyme: Schrittweite.

Siehe auch: ML, Hypothese, Parameter, Gradientenabstieg, stochastischer Gradientenabstieg (SGD), projected gradient descent (projected GD), Schrittweite, ERM, lineares Modell, Hyperparameter, parameter space, Hypothesenraum.

Querverweise

Literatur

1. Boyd and Vandenberghe (2004). Convex Optimization. Cambridge Univ. Press, Cambridge, U.K.

2. Nesterov (2004). Introductory Lectures on Convex Optimization: A Basic

Course. Kluwer Academic, Boston, MA, USA.