logistische Regression

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

Die logistische Regression lernt eine lineare Hypothese-Abbildung für binäre Klassifikation, also für Datenpunkte, deren Label einen von zwei Werten annimmt, $y \in \{-1, 1\}$. Das vorhergesagte Label ist das Vorzeichen des Hypothese-Werts, und dessen Absolutwert misst die Sicherheit der Klassifikation. Die Parameter werden durch empirische Risikominimierung (ERM) mit dem mittleren logistische Verlustfunktion auf dem Trainingsdatensatz gelernt. Die Minimierung des mittleren logistische Verlustfunktion ist äquivalent zur maximum likelihood-Schätzung unter einem einfachen Wahrscheinlichkeitsmodell, bei dem das Sigmoidfunktion des Hypothese-Werts die Wahrscheinlichkeit für das Label 1 ist. Trägt ein Datenpunkt diese Wahrscheinlichkeit als soft label statt einer Klasse, wird der logistische Verlustfunktion durch die cross-entropy zwischen soft label und Vorhersage, äquivalent deren Kullback–Leibler divergence (KL divergence), ersetzt. Die resultierende Zielfunktion ist konvex und smooth, sodass Varianten von Gradientenabstieg zu ihrer Minimierung eingesetzt werden können.

Definition

Man betrachte die Vorhersage, ob es morgen friert, anhand aktueller Temperaturmessungen. Aufzeichnungen einer Wetterstation machen daraus Daten: Jeder Tag ist ein Datenpunkt mit zwei numerischen Merkmale, der Lufttemperatur um 15:00 Uhr (UTC) des Tages und der des Vortags, zusammengefasst im Merkmalsvektor $\mathbf{x} = (x_1, x_2)^\top \in \mathbb{R}^2$. Das Label $y \in \{-1, 1\}$ ist $y = 1$, wenn die Messung am Folgetag um 06:00 Uhr (UTC) mindestens 0°C beträgt, und $y = -1$ sonst.

Abb. 1 zeigt solche Datenpunkte. Die logistische regression lernt für eine solche binäre Klassifikation-Aufgabe eine lineare Hypothese-Abbildung $h(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$ mit Gewicht-Vektor $\mathbf{w}$ und Offset b (Bishop, 2006, Sect. 4.3.2; Hastie et al., 2009, Sect. 4.4; Shalev-Shwartz and Ben-David, 2014, Sect. 9.3). Trotz ihres Namens löst die Methode ein Klassifikation-Problem: Das regression bezieht sich auf das Fitten der reellwertigen Abbildung h, nicht auf ein numerisches Label. Die Menge der Merkmalsvektoren mit $h(\mathbf{x}) = 0$ ist die Entscheidungsgrenze zwischen den beiden Klassen, und $\mathbf{w}$ ist ihr Normalenvektor.

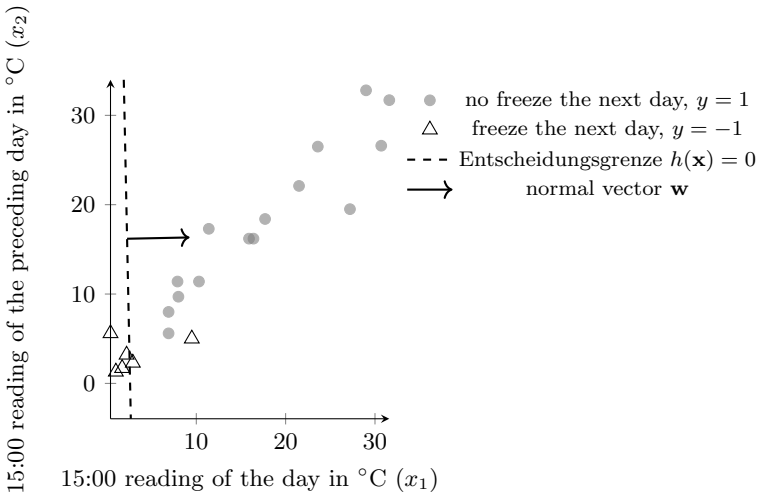

Abbildung 1: Datenpunkte für die Frostvorhersage an der Wetterstation Krems an der Donau: Jeder Tag ist bei der 15:00 (UTC)-Messung des Tages (x_1) und des Vortages (x_2) platziert. Tage, denen ein Frosttag folgt (Messung um 06:00 (UTC) unter 0°C), sind als offene Dreiecke gezeichnet; sie konzentrieren sich dort, wo beide Merkmale klein sind, aber die beiden Klassen überlappen. Einundzwanzig der 361 Tage sind gezeichnet, jeder 23. Tag ohne und jeder 6. Tag mit einem Frostnachfolger. Die gestrichelte Linie ist die Entscheidungsgrenze der aus allen 361 Tagen gelernten Hypothese und der durchgezogene Pfeil ihr Normalenvektor $\mathbf{w}$. Daten generiert von `pythondemos/logreg.py`

Die Vorhersage, also die Klassifikation, ergibt sich aus dem Vorzeichen von $h(\mathbf{x})$, womit die gelernte h eine linearer Klassifikator ist, deren Entscheidungsgrenze die Hyperebene $\mathbf{w}^\top \mathbf{x} + b = 0$ mit Normalenvektor $\mathbf{w}$ ist. Der Absolutwert $|h(\mathbf{x})|$ dient als Maß für die Sicherheit der Klassifikation: Er wächst mit dem Abstand von $\mathbf{x}$ zur Entscheidungsgrenze, und ein Wert nahe null zeigt geringe Sicherheit an.

Die Parameter $\mathbf{w}$ und b der logistischen Regression werden mittels empirische Risikominimierung (ERM) gelernt. Die dabei minimierte Verlust ist der logistische Verlustfunktion. Für einen Datenpunkt $(\mathbf{x}, y)$ verursacht eine Hypothese h die Kosten

$$L\left((\mathbf{x}, y), h\right) = \ln\left(1 + \exp(-y\, h(\mathbf{x}))\right).$$

Abb. 2 zeigt diese Verlust für einen Datenpunkt mit $y = 1$ als Funktion von $h(\mathbf{x})$. Eine korrekte Klassifikation mit hoher Sicherheit verursacht eine geringe Verlust. Eine falsche Klassifikation verursacht eine Verlust, deren Größe mit der Sicherheit in die falsche Richtung wächst.

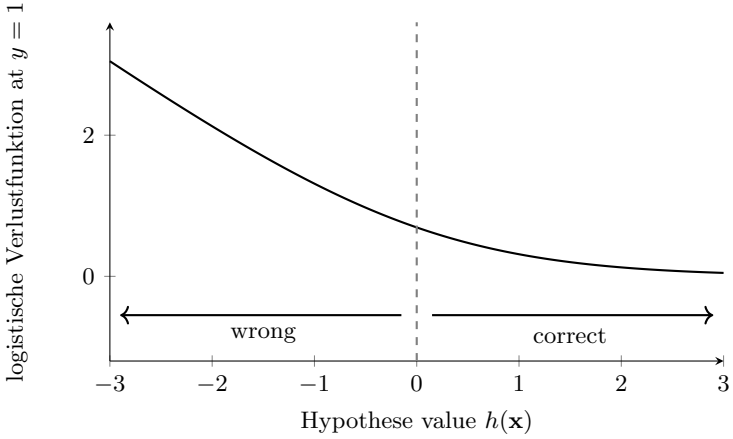

Abbildung 2: Der logistische Verlustfunktion, der bei einem Datenpunkt mit Label $y = 1$ anfällt, als Funktion des Hypothese-Wertes $h(\mathbf{x})$. Eine korrekte Klassifikation, $h(\mathbf{x}) > 0$, mit hoher Konfidenz verursacht einen kleinen Verlust; der Verlust wächst, wenn $h(\mathbf{x})$ negativ wird, d.h. wenn die Konfidenz in eine falsche Klassifikation zunimmt. Die gestrichelte Linie markiert die Entscheidungsgrenze $h(\mathbf{x}) = 0$, und die Pfeile zeigen auf wachsende Konfidenz in eine korrekte (rechts) oder falsche (links) Klassifikation. Daten generiert von `pythondemos/logreg.py`

Durch Mittelung dieser Verlust über das Trainingsdatensatz $\mathcal{D}^{(\text{train})} = \{(\mathbf{x}^{(r)}, y^{(r)})\}_{r=1}^m$ aus m Datenpunkte ergibt ERM

$$(\widehat{\mathbf{w}}, \widehat{b}) \in \underset{\mathbf{w}\in\mathbb{R}^d,\, b\in\mathbb{R}}{\arg\min}\ f(\mathbf{w}, b), \quad \text{mit}$$
$$f(\mathbf{w}, b) := \frac{1}{m}\sum_{r=1}^{m} \ln\Big(1 + \exp\big(-y^{(r)}(\mathbf{w}^\top \mathbf{x}^{(r)} + b)\big)\Big).$$

Der logistische Verlustfunktion ist der negative Log-Likelihood des Label unter einem einfachen Wahrscheinlichkeitsmodell für die Klassifikation-Aufgabe.

In diesem Wahrscheinlichkeitsmodell ist, gegeben das Label y, der Merkmalsvektor $\mathbf{x}$ eine Gaußsche Zufallsvariable, deren Mittelwert $\boldsymbol{\mu}^{(y)}$ vom Label abhängt, während ihre Kovarianzmatrix $\mathbf{C}$ unabhängig ist; die beiden Mittelwerte $\boldsymbol{\mu}^{(-1)}, \boldsymbol{\mu}^{(+1)} \in \mathbb{R}^d$ sind die Parameter des Modells. Sind beide Labels a priori gleich wahrscheinlich, so ist die linearer Klassifikator mit dem geringsten Risiko unter diesem Modell, der Bayes-Klassifikator, derjenige, der $\mathbf{x}$ dem Label mit dem näheren Mittelwert zuordnet, wobei die Abstände durch $\mathbf{C}^{-1}$ gemessen werden, und dessen Entscheidungsgrenze die Hyperebene durch den Mittelpunkt der beiden Mittelwerte ist (Bishop, 2006, Sect. 4.2.1). Sein Gewicht-Vektor ist $\mathbf{w} = \mathbf{C}^{-1}\big(\boldsymbol{\mu}^{(+1)} - \boldsymbol{\mu}^{(-1)}\big)$. Sein Offset ist $b = -\frac{1}{2}\big(\boldsymbol{\mu}^{(+1)} + \boldsymbol{\mu}^{(-1)}\big)^\top \mathbf{w}$, was die Entscheidungsgrenze auf den Mittelpunkt legt, zuzüglich des Logarithmus des

Verhältnisses der beiden a-priori-Wahrscheinlichkeiten, falls diese unterschiedlich sind. Die Multiplikation von $\mathbf{w}$ und b mit derselben positiven Zahl lässt die Entscheidungsgrenze unverändert, aber nur dieses Paar reproduziert die nachfolgende bedingte Wahrscheinlichkeit. Nach dem Modell ist die bedingte Wahrscheinlichkeit für $y = 1$ das Sigmoidfunktion $\sigma(\mathbf{w}^\top \mathbf{x} + b)$ mit $\sigma(z) = 1/(1 + e^{-z})$ (Bishop, 2006, Sect. 4.2.1, Gl. (4.65)–(4.67)). Die logistische Regression schätzt $\mathbf{w}$ und b direkt, indem sie die bedingte likelihood function der Labels gegeben die Merkmalsvektoren maximiert. Damit bleibt die Verteilung der Merkmalsvektoren unbestimmt und es handelt sich um einen anderen Schätzer als die maximum likelihood-Schätzung für die Mittelwerte und die Kovarianzmatrix des Gauß-Modells (Hastie et al., 2009, Sect. 4.4.5). Die Maximierung der bedingten likelihood function entspricht der Minimierung des mittleren logistische Verlustfunktion f (Murphy, 2012, Sect. 8.3.1).

Das Wahrscheinlichkeitsmodell macht die oben genannte Sicherheitseinschätzung $|h(\mathbf{x})|$ präzise: $\sigma(h(\mathbf{x}))$ ist die Wahrscheinlichkeit, dass das Label eines Datenpunkt mit Merkmalsvektor $\mathbf{x}$ $y = 1$ ist, und $\sigma(-h(\mathbf{x})) = 1 - \sigma(h(\mathbf{x}))$ die Wahrscheinlichkeit für $y = -1$. Eine Prognose $h(\mathbf{x}) = 2$ besagt also, dass 88 Prozent der Tage mit diesen Temperaturen frostfrei bleiben, und eine Prognose von 0, dass es bei der Hälfte der Tage so ist. Ob diese Zahlen vertrauenswürdig sind, ist die Frage der calibration (Dawid, 1982).

In den meisten Anwendungen trägt ein Datenpunkt nur sein Label, also die eingetretene Klasse. Der logistische Verlustfunktion ist die Verlust für diesen Fall. Er ist die cross-entropy zwischen der auf das beobachtete Label konzentrierten probability mass function (pmf) und der vorhergesagten pmf $(\sigma(h(\mathbf{x})), 1 - \sigma(h(\mathbf{x})))$ (Bishop, 2006, Sect. 4.3.2). Manche Datenpunkte tragen mehr als eine Klasse: ein soft label $\tilde{y} \in [0, 1]$, also die Wahrscheinlichkeit für $y = 1$ selbst. Dies kann die Häufigkeit von Frost unter den aufgezeichneten Tagen mit denselben Temperaturen sein, der Anteil mehrerer Annotatoren, die die Klasse gewählt haben, oder die Ausgabe eines größeren Klassifikator, wie bei der knowledge distillation. Für einen solchen Datenpunkt wird der logistische Verlustfunktion durch die cross-entropy zwischen dem soft label und der Vorhersage ersetzt,

$$L\left((\mathbf{x}, \tilde{y}), h\right) = -\tilde{y} \ln \sigma(h(\mathbf{x})) - (1 - \tilde{y}) \ln \big(1 - \sigma(h(\mathbf{x}))\big),$$

die für $\tilde{y} \in \{0, 1\}$ wieder dem logistische Verlustfunktion entspricht. Das Lernen minimiert dann wie zuvor den Mittelwert dieser Verlust über das Trainingsdatensatz.

Diese cross-entropy entspricht der Kullback–Leibler divergence (KL divergence) von der vorhergesagten pmf zur soft label-pmf $(\tilde{y}, 1 - \tilde{y})$ plus der entropy der soft label-pmf (Bishop, 2006, Sect. 1.6.1, Gl. (1.113)). Mit $q := \sigma(h(\mathbf{x}))$ gilt nämlich

$$-\tilde{y} \ln q - (1 - \tilde{y}) \ln(1 - q) = \tilde{y} \ln \frac{\tilde{y}}{q} + (1 - \tilde{y}) \ln \frac{1 - \tilde{y}}{1 - q} - \tilde{y} \ln \tilde{y} - (1 - \tilde{y}) \ln(1 - \tilde{y}),$$

wobei die ersten beiden Summanden die KL divergence und die letzten beiden die entropy sind, die nicht von h abhängt. Die Minimierung beider lernt daher

dieselbe h, und die Verlust auf einem Datenpunkt ist genau dann am kleinsten, wenn $q = \tilde{y}$ ist, also die Sicherheit der Vorhersage der Wahrscheinlichkeit der Klasse entspricht. Ein hartes Label kann so nie exakt getroffen werden, da σ nie 0 oder 1 erreicht. Der logistische Verlustfunktion sinkt daher weiter, je größer $|h(\mathbf{x})|$ auf einem korrekt klassifizierten Datenpunkt wird.
Bei harten oder weichen Labels werden die Parameter durch Minimierung des mittleren Verlust f gefunden. Im Gegensatz zur lineare Regression mit dem squared error loss lässt die Minimierung keine geschlossene Lösung zu (Bishop, 2006, Sect. 4.3.3). Wie bei der lineare Regression ist die Zielfunktion f jedoch konvex (Murphy, 2012, Sect. 8.3.1, Eq. (8.7)) und smooth, sodass Gradientenabstieg anwendbar ist. Die Parameter werden als Paar $(\mathbf{w}, b)$ zusammengefasst, und der Gradientenabstieg-Schritt lautet

$$(\mathbf{w}, b)^{(t+1)} = T\big((\mathbf{w}, b)^{(t)}\big), \text{ mit } T(\mathbf{w}, b) := (\mathbf{w}, b) - \eta \nabla f(\mathbf{w}, b), \text{ für } t = 0, 1, \ldots,$$

mit der Lernrate η. Der Gradient ∇f des mittleren logistische Verlustfunktion hat den Gewichtsanteil

$$\nabla_{\mathbf{w}} f = -\frac{1}{m} \sum_{r=1}^{m} y^{(r)} \mathbf{x}^{(r)} \, \sigma\big(- y^{(r)} (\mathbf{w}^\top \mathbf{x}^{(r)} + b)\big),$$

und analog für b.

Das Paar $(\mathbf{w}, b)$ ist genau dann ein Fixpunkt des Operator T, wenn $\nabla f(\mathbf{w}, b) = \mathbf{0}$, was aufgrund der Konvexität von f genau dann der Fall ist, wenn $(\mathbf{w}, b)$ ein Minimierer ist. Der Gradient ∇f ist Lipschitz continuous, da die Ableitung $\sigma'(z) = \sigma(z)(1 - \sigma(z)) \leq 1/4$ der Sigmoidfunktion beschränkt ist: die Hessian von f ist $\frac{1}{m} \sum_{r=1}^{m} s^{(r)} \tilde{\mathbf{x}}^{(r)} \big(\tilde{\mathbf{x}}^{(r)}\big)^\top$ mit $\tilde{\mathbf{x}}^{(r)} := (\mathbf{x}^{(r)}, 1)$ und $s^{(r)} := \sigma\big(h(\mathbf{x}^{(r)})\big) \, \sigma\big(- h(\mathbf{x}^{(r)})\big) \in (0, 1/4]$, sodass eine Konstante $L = \frac{1}{4m} \sum_{r=1}^{m} \big\|\tilde{\mathbf{x}}^{(r)}\big\|_2^2$ ist. Für eine konstante Lernrate $\eta < 2/L$ verringert jeder Gradientenabstieg-Schritt f (Beck, 2017, Lemma 5.7). Für die Standardwahl $\eta = 1/L$ und einen Minimierer $(\widehat{\mathbf{w}}, \widehat{b})$ von f beträgt der Abstand $f\big((\mathbf{w}, b)^{(t)}\big) - \min f$ nach $t \geq 1$ Schritten höchstens $2L \left\|(\mathbf{w}, b)^{(0)} - (\widehat{\mathbf{w}}, \widehat{b})\right\|_2^2 / t$ (Bubeck, 2015, Thm. 3.3), wobei die Iterationszählung hier bei 1 beginnt, während die obige bei 0 beginnt. Die Iterierten $(\mathbf{w}, b)^{(t)}$ selbst müssen nicht konvergieren: Auf einem linear separierbaren Trainingsdatensatz wird das minimum nicht erreicht (Hastie et al., 2009, Sect. 4.4.5), und die Norm der Iterierten wächst unbegrenzt, während der Trainingsfehler gegen Null geht. Das Hinzufügen des Strafterm $\alpha \|\mathbf{w}\|_2^2$ mit einem positiven α zu f beseitigt diese Singularität (Bishop, 2006, Sect. 4.3.2). Die penalisierten Zielfunktion ist strictly convex in $(\mathbf{w}, b)$: ihre zweite Ableitung entlang einer Richtung $(\mathbf{u}, c) \neq (\mathbf{0}, 0)$ ist

$$\frac{1}{m} \sum_{r=1}^{m} s^{(r)} \big(\mathbf{u}^\top \mathbf{x}^{(r)} + c\big)^2 + 2\alpha \|\mathbf{u}\|_2^2,$$

mit dem positiven $s^{(r)}$ von oben, und dies verschwindet nur für $\mathbf{u} = \mathbf{0}$ und $c = 0$. Wenn beide Labels im Trainingsdatensatz vorkommen, wächst die penalisierten

Zielfunktion auch entlang jeder unbeschränkten Folge von Paaren $(\mathbf{w}, b)$ unbegrenzt, sodass sie einen eindeutigen Minimierer besitzt. Ihr Gradient ist Lipschitz continuous mit der Konstanten $L + 2\alpha$. Für eine Lernrate $\eta < 2/(L + 2\alpha)$ konvergieren die Werte der Gradientenabstieg-Iterierten wieder zum Minimum, die Iterierten bleiben beschränkt und nähern sich dem eindeutigen Minimierer.

In der Praxis wird einfaches Gradientenabstieg selten zur Minimierung von f verwendet: Die smooth konvex Zielfunktion wird schneller durch die Newton's method, äquivalent iterativ reweighted least squares (Hastie et al., 2009, Sect. 4.4.1, Eq. (4.28)), durch den Quasi-Newton-Löser L-BFGS (Liu and Nocedal, 1989) und durch coordinate descent (Wright, 2015) minimiert.

Was diese Löser minimieren, ist der mittlere logistische Verlustfunktion, aber das ist nicht der einzige Verlust, mit dem eine linearer Klassifikator gelernt werden kann. Im Prinzip dient auch der squared error loss: Mit den als reelle Zahlen $+1$ und -1 kodierten Label-Werten ist der quadratische Fehler $\big(y - (\mathbf{w}^\top \mathbf{x} + b)\big)^2$ wohldefiniert. Die Minimierung des squared error loss hat jedoch einen Fehler: Sie bestraft eine Vorhersage $\mathbf{w}^\top \mathbf{x} + b$, die groß, positiv und korrekt ist, während der logistische Verlustfunktion weiter abnimmt, wenn diese Margin wächst (Abb. 3). Der hinge loss der Stützvektormaschine (SVM) zeigt dieses Verhalten bei einem korrekt klassifizierten Datenpunkt, bleibt aber flach, sobald die Margin eins überschreitet.

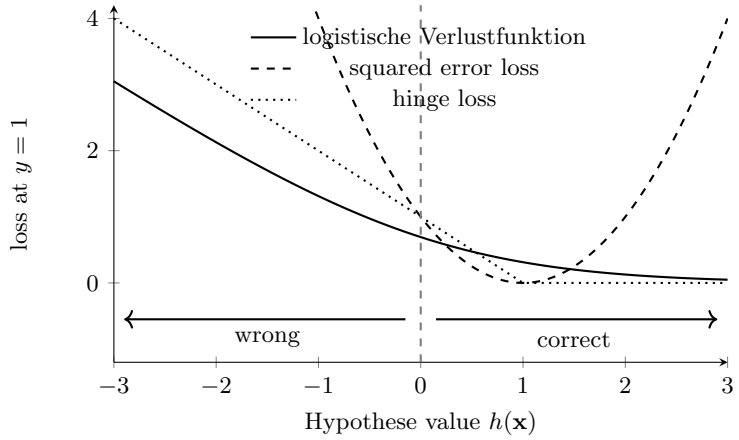

Abbildung 3: Der logistische Verlustfunktion, squared error loss und hinge loss, die bei einem Datenpunkt mit Label $y = 1$ anfallen, als Funktion des Hypothese-Wertes $h(\mathbf{x})$. Der squared error loss steigt an und bestraft eine zuversichtliche korrekte Vorhersage, während der logistische Verlustfunktion weiter abnimmt; der hinge loss bleibt flach, sobald die Margin eins überschreitet. Die gestrichelte Linie markiert die Entscheidungsgrenze $h(\mathbf{x}) = 0$, und die Pfeile zeigen auf wachsende Konfidenz in eine korrekte (rechts) oder falsche (links) Klassifikation. Daten generiert von `pythondemos/logreg.py`

Für mehr als zwei Label-Werte lässt sich die logistische Regression auf zwei Arten erweitern. Eine einfache Reduktion trainiert eine binäre Klassifikation logistische Regression pro Label, wobei jede dieses Label vom Rest trennt, und prognostiziert mit der zuversichtlichsten von ihnen; dies ist die Eins-gegen-Rest-Klassifikation. Eine Alternative ersetzt die Sigmoidfunktion durch die softmax function, die die Wahrscheinlichkeiten aller Labels gemeinsam als eine pmf modelliert, die sich zu eins summiert; das Lernen ihrer Parameter mittels maximum likelihood führt zur multinomialen logistischen Regression (Bishop, 2006, Sect. 4.3.4).

Siehe auch: Klassifikation, binäre Klassifikation, linearer Klassifikator, Entscheidungsgrenze, Sigmoidfunktion, logistische Verlustfunktion, squared error loss, hinge loss, ERM, Gradientenabstieg, maximum likelihood, lineare Regression, softmax function, Regularisierung, calibration, soft label, cross-entropy, KL divergence.

Querverweise

Literatur

1. Beck (2017). First-Order Methods in Optimization. SIAM-Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

2. Bishop (2006). Pattern Recognition and Machine Learning. Springer Science+Business Media, New York, NY, USA.

3. Bubeck (2015). Convex Optimization: Algorithms and Complexity. Found. Trends Mach. Learn., 8(3–4), pp. 231–357.

4. Dawid (1982). The Well-Calibrated Bayesian. Journal of the American Statistical Association, 77(379), pp. 605–610.

5. Liu and Nocedal (1989). On the Limited Memory Method for Large Scale Optimization. Mathematical Programming B, 5(3), pp. 503–528.

6. Murphy (2012). Machine Learning: A Probabilistic Perspective. MIT Press, Cambridge, MA, USA.

7. Shalev-Shwartz and Ben-David (2014). Understanding Machine Learning: From Theory to Algorithms. Cambridge Univ. Press, New York, NY, USA.

8. Wright (2015). Coordinate Descent Algorithms. Mathematical Programming, 151(1), pp. 3–34.

9. Hastie et al. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd ed., Springer Science+Business Media, New York, NY, USA.