linear regression

Authors

Alexander Jung*, Department of Computer Science, Aalto University, Espoo, Finland

*Corresponding author: alex.jung@aalto.fi

Abstract

Linear regression is a regression method that learns a linear hypothesis map for predicting the numeric label of a data point from its features. Its least squares variant chooses a map that minimizes the average squared error loss on a training set. Linear regression is an instance of empirical risk minimization (ERM) that is obtained by using the linear model and the squared error loss. The optimal model parameters are determined by the normal equations. This system can be solved in closed form or iteratively, for example, by gradient descent (GD). Adding a penalty term to the average squared error loss yields regularized variants such as ridge regression and least absolute shrinkage and selection operator (Lasso).

Definition

Linear regression methods learn a linear hypothesis map that delivers a prediction of the numeric label of a data point. The prediction is based solely on the features of the data point, which are fed as input to the learned hypothesis map. Linear regression can be used to predict tomorrow's temperature from weather measurements recorded over the last five days, using them as features and tomorrow's temperature as the label (Jung, 2022, Sect. 2.1).

Formally, linear regression learns a linear hypothesis map $h^{(\mathbf{w})}(\mathbf{x}) = \mathbf{w}^{\top}\mathbf{x}$ to predict the numeric label $y \in \mathbb{R}$ of a data point from its feature vector $\mathbf{x} = \big(x_1, \ldots, x_d\big)^{\top} \in \mathbb{R}^d$. Here, d denotes the number of features of a data point. The model parameters $\mathbf{w}$ are learned from a training set $\mathcal{D}^{(\text{train})} = \big\{ \big(\mathbf{x}^{(r)}, y^{(r)}\big) \big\}_{r=1}^{m}$ of data points. Appending a constant feature to $\mathbf{x}$ lets one entry of $\mathbf{w}$ act as an intercept, so $\mathbf{w}^{\top}\mathbf{x}$ represents an affine function of the original measurements. In what follows, $\mathbf{x}$ denotes the feature vector that is fed to the linear hypothesis map. It can be either the original or the augmented feature vector.

The least squares variant of linear regression measures the quality of a linear hypothesis map by the average squared error loss on the training set. As an instance of empirical risk minimization (ERM), it learns the model parameters $\mathbf{w}$ by solving the optimization problem

$$\min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{m} \sum_{r=1}^{m} \big(y^{(r)} - \mathbf{w}^{\top}\mathbf{x}^{(r)}\big)^2.$$

Fig. 1 illustrates this linear regression problem for a training set with a constant scalar feature $x^{(r)} = 1$ for $r = 1, \ldots, m$.

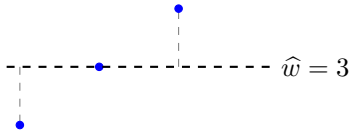

Figure 1: For a linear model with $d = 1$ and the constant feature $x = 1$ for every data point (blue filled circles), linear regression reduces to computing the average $\widehat{w} = (1/m) \sum_{r=1}^{m} y^{(r)}$ of the labels: this average minimizes the average squared error loss. The horizontal dashed line sits at the height $\widehat{w}$, and the vertical gray segments are the errors $y^{(r)} - \widehat{w}$ entering that loss

The optimization problem can be written more compactly using the feature matrix $\mathbf{X}$ and the label vector $\mathbf{y}$,

$$\mathbf{X} = \left(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\right)^\top \in \mathbb{R}^{m \times d}, \qquad \mathbf{y} = \left(y^{(1)}, \ldots, y^{(m)}\right)^\top \in \mathbb{R}^{m}.$$

In terms of $\mathbf{X}$ and $\mathbf{y}$, the optimization problem reads

$$\min_{\mathbf{w} \in \mathbb{R}^d} \underbrace{\frac{1}{m} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2}_{f(\mathbf{w})}.$$

The average squared error loss $f(\mathbf{w})$ is a convex and differentiable function of $\mathbf{w}$. By the zero-gradient condition (Boyd and Vandenberghe, 2004, Sect. 4.2.3), a vector $\widehat{\mathbf{w}}$ solves this optimization problem if and only if it satisfies the system of linear equations (Strang, 2016)

$$\mathbf{X}^\top \mathbf{X} \widehat{\mathbf{w}} = \mathbf{X}^\top \mathbf{y}. \tag{1}$$

The normal equations (1) always have a solution because $\mathbf{X}^\top \mathbf{y}$ lies in the column space of $\mathbf{X}^\top \mathbf{X}$ (Golub and Van Loan, 2013, Sect. 5.5).

However, the solution of (1) is unique only if the feature matrix $\mathbf{X}$ has full column rank. This requires at least as many data points as features, $m \geq d$. When this holds, and the columns of $\mathbf{X}$ are linearly independent, the matrix $\mathbf{X}^\top \mathbf{X}$ is invertible and (1) has the unique closed-form solution $\widehat{\mathbf{w}} = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{y}$.

If instead $m < d$ (more features than data points), $\mathbf{X}$ cannot have full column rank, $\mathbf{X}^\top \mathbf{X}$ is singular, and the optimization problem has infinitely many solutions. All of them incur the same minimal average squared error loss on the training set, but their predictions for data points outside the training set can differ arbitrarily. Choosing among them is therefore a matter of generalization (see overfitting). A unique solution can be selected, for example, by picking the minimum-norm solution $\mathbf{X}^+ \mathbf{y}$ given by the pseudoinverse $\mathbf{X}^+$. A common remedy is regularization: adding a penalty term to the average squared error loss. The penalty term can be interpreted as an estimate of how much higher

the loss is on data points outside the training set than on it (Hastie et al., 2009, Ch. 7). This yields regularized variants of linear regression, with ridge regression using the penalty term $\alpha\|\mathbf{w}\|_2^2$ and the least absolute shrinkage and selection operator (Lasso) using the penalty term $\alpha\|\mathbf{w}\|_1$.

Linear regression also has a statistical interpretation. Consider a probabilistic model with a joint probability distribution $\mathbb{P}^{(\mathbf{x},y)}$ over the features and the label. Under the squared error loss, the Bayes estimator (i.e., the hypothesis with minimum risk) is the conditional expectation $\mathbb{E}\{y \mid \mathbf{x}\}$ (Lehmann and Casella, 1998, Ch. 4; Papoulis and Pillai, 2002). In this statistical interpretation, $\mathbf{x}$ denotes the original feature vector. When the features and the label are jointly Gaussian random variables (Gaussian RVs) with zero mean and invertible covariance matrix $\mathbf{C}^{(\mathbf{x})} = \mathbb{E}\{\mathbf{x}\mathbf{x}^\top\}$, this Bayes estimator is

$$h^*(\mathbf{x}) = \big(\mathbf{w}^\star\big)^\top \mathbf{x}, \text{ with } \mathbf{w}^\star = \big(\mathbf{C}^{(\mathbf{x})}\big)^{-1}\,\mathbf{c}^{(\mathbf{x},y)}.$$

Here, $\mathbf{c}^{(\mathbf{x},y)} = \mathbb{E}\{\mathbf{x}y\}$ is the covariance between the features $\mathbf{x}$ and the label y. This Bayes estimator is linear in $\mathbf{x}$. Without the zero-mean assumption, it is an affine function of $\mathbf{x}$. However, as for the linear hypothesis map above, the intercept can be absorbed by appending a constant feature. Least-squares linear regression is the ERM counterpart of this predictor: it replaces $\mathbf{C}^{(\mathbf{x})}$ and $\mathbf{c}^{(\mathbf{x},y)}$ with the sample-based estimates $(1/m)\mathbf{X}^\top\mathbf{X}$ and $(1/m)\mathbf{X}^\top\mathbf{y}$, recovering, for invertible $\mathbf{X}^\top\mathbf{X}$, the unique solution $\widehat{\mathbf{w}} = \big(\mathbf{X}^\top\mathbf{X}\big)^{-1}\mathbf{X}^\top\mathbf{y}$ of (1).

Instead of solving the normal equations (1) directly (via the inverse matrix or pseudoinverse), machine learning (ML) methods often solve it by an iterative optimization method, as implemented in widely used ML software libraries such as scikit-learn (Pedregosa et al., 2011) and PyTorch (Paszke et al., 2019). Starting from initial model parameters $\mathbf{w}^{(0)}$, such a method repeatedly applies an update operator $\mathcal{F}$,

$$\mathbf{w}^{(t+1)} = \mathcal{F}\big(\mathbf{w}^{(t)}\big), \text{ for } t = 0, 1, \ldots.$$

The operator $\mathcal{F}$ is designed such that the sequence of model parameters $\mathbf{w}^{(t)}$ converges to a solution of (1).

One important example of such an iterative optimization method is gradient descent (GD), which uses the GD step operator

$$\mathcal{F}^{(\eta)}(\mathbf{w}) = \mathbf{w} - \eta\nabla f(\mathbf{w}).$$

Here, $\eta > 0$ is a step size that must be chosen small enough to ensure convergence (Bertsekas, 2016, Sect. 1.2). The superscript in $\mathcal{F}^{(\eta)}$ makes the dependence of the operator on η explicit. Inserting the average training error into the general form of the GD step yields the explicit update

$$\begin{aligned}\mathcal{F}^{(\eta)}(\mathbf{w}) &= \mathbf{w} + \frac{2\eta}{m}\sum_{r=1}^{m}\big(y^{(r)} - \mathbf{w}^\top\mathbf{x}^{(r)}\big)\mathbf{x}^{(r)} \\ &= \Big(\mathbf{I} - \frac{2\eta}{m}\mathbf{X}^\top\mathbf{X}\Big)\mathbf{w} + \frac{2\eta}{m}\mathbf{X}^\top\mathbf{y}.\end{aligned}$$

The fixed points of $\mathcal{F}^{(\eta)}$ are exactly the solutions of the normal equations (1), since $\mathcal{F}^{(\eta)}(\widehat{\mathbf{w}}) = \widehat{\mathbf{w}}$ holds if and only if $\mathbf{X}^\top\mathbf{X}\widehat{\mathbf{w}} = \mathbf{X}^\top\mathbf{y}$. The update operator is affine,

$$\mathcal{F}^{(\eta)}(\mathbf{w}) = \mathbf{M}\mathbf{w} + \mathbf{b},$$

with linear part $\mathbf{M} = \mathbf{I} - (2\eta/m)\mathbf{X}^\top\mathbf{X}$ and offset $\mathbf{b} = (2\eta/m)\,\mathbf{X}^\top\mathbf{y}$.

When $\mathbf{X}$ has full column rank, so that $\mathbf{X}^\top\mathbf{X}$ is positive definite, $\mathcal{F}^{(\eta)}$ is a contractive operator with respect to the Euclidean norm for every step size $0 < \eta < m/\lambda_{\max}$ (Nesterov, 2004, Thm. 2.1.14). Here, $\lambda_{\max}$ is the largest eigenvalue of $\mathbf{X}^\top\mathbf{X}$, and the fixed-point iteration converges to the unique solution $\widehat{\mathbf{w}}$. Moreover, the convergence speed of GD is governed by the ratio of the largest to the smallest eigenvalue of $\mathbf{X}^\top\mathbf{X}$, that is, by its condition number $\kappa\big(\mathbf{X}^\top\mathbf{X}\big) = \lambda_{\max}/\lambda_{\min}$ (Boyd and Vandenberghe, 2004, Sect. 9.3).

The GD step above uses the entire training set to compute the gradient. When the data points instead arrive sequentially at time instants $t = 1, 2, \ldots$, or the training set is too large to fit in memory, the model parameters can be learned by online gradient descent (online GD). At each time step t, it applies a single GD step using only the arriving data point $\big(\mathbf{x}^{(t)}, y^{(t)}\big)$,

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + 2\eta\big(y^{(t)} - \big(\mathbf{w}^{(t)}\big)^\top\mathbf{x}^{(t)}\big)\mathbf{x}^{(t)}.$$

This is the GD step on the loss function $\big(y^{(t)} - \mathbf{w}^\top\mathbf{x}^{(t)}\big)^2$ of the arriving data point alone, rather than on the average squared error loss $f(\mathbf{w})$. It is the least mean squares (LMS) update (Bishop, 2006, Sect. 3.1.3). It avoids storing the entire training set, and its per-iteration complexity is independent of the training set size m.

The optimality condition (1) allows one to study the stability of linear regression. Ideally, the learned model parameters are insensitive to limited perturbations of the training set. A label-only perturbation, for example, replaces a single label of the training set with an outlier or another corrupted value. A perturbed training set yields a feature matrix $\widetilde{\mathbf{X}} = \mathbf{X} + \Delta\mathbf{X}$ and a label vector $\widetilde{\mathbf{y}} = \mathbf{y} + \Delta\mathbf{y}$, with perturbation matrix $\Delta\mathbf{X}$ and vector $\Delta\mathbf{y}$. This gives the perturbed normal equations

$$\widetilde{\mathbf{X}}^\top\widetilde{\mathbf{X}}\widetilde{\mathbf{w}} = \widetilde{\mathbf{X}}^\top\widetilde{\mathbf{y}}. \tag{2}$$

Matrix perturbation theory quantifies how much $\widetilde{\mathbf{w}}$ deviates from a solution $\widehat{\mathbf{w}}$ of the clean normal equations (1) (Golub and Van Loan, 2013, Sect. 2.6).

For a label-only perturbation, $\Delta\mathbf{X} = \mathbf{0}$, the perturbed and clean normal equations share the coefficient matrix $\mathbf{X}^\top\mathbf{X}$. Assume $\mathbf{X}$ has full column rank, so $\mathbf{X}^\top\mathbf{X}$ is invertible and the solution is unique. Subtracting the clean normal equations (1) from the perturbed normal equations (2) yields

$$\widetilde{\mathbf{w}} - \widehat{\mathbf{w}} = \big(\mathbf{X}^\top\mathbf{X}\big)^{-1}\mathbf{X}^\top\Delta\mathbf{y} = \mathbf{X}^+\Delta\mathbf{y},$$

with the pseudoinverse $\mathbf{X}^+$. This in turn allows one to quantify the effect of perturbing the training set as

$$\|\widetilde{\mathbf{w}} - \widehat{\mathbf{w}}\|_2 \le \big\|\mathbf{X}^+\big\|_2\,\|\Delta\mathbf{y}\|_2.$$

When $\mathbf{X}$ has full column rank, $\|\mathbf{X}^+\|_2 = 1/\sqrt{\lambda_{\min}(\mathbf{X}^\top\mathbf{X})} = \sqrt{\kappa(\mathbf{X}^\top\mathbf{X})}/\|\mathbf{X}\|_2$, where $\kappa\big(\mathbf{X}^\top\mathbf{X}\big) = \lambda_{\max}/\lambda_{\min}$ is the condition number of $\mathbf{X}^\top\mathbf{X}$. In the setting of Fig. 1, $\mathbf{X} = \mathbf{1}$ (the all-ones vector), so $\mathbf{X}^+\Delta\mathbf{y} = (1/m)\sum_{r=1}^{m}\Delta y^{(r)}$. Fig. 2 illustrates such a label-only perturbation of the training set from Fig. 1: the single label perturbation $\Delta y^{(3)} = 6$ gives $\widetilde{w} - \widehat{w} = 6/3 = 2$, i.e., the outlier pulls the average of the labels upward, from $\widehat{w} = 3$ to $\widetilde{w} = 5$.

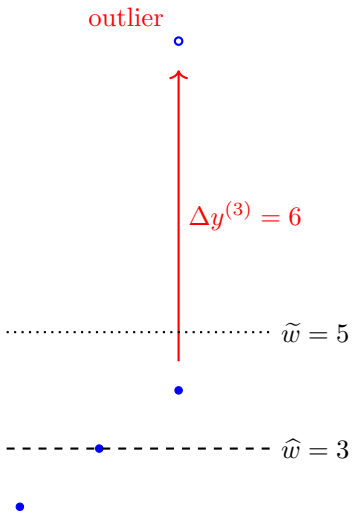

Figure 2: The perturbation $\Delta y^{(3)} = 6$ (red arrow) replaces the third label of the training set from Fig. 1 with an outlier: filled circles denote the original data points, as in Fig. 1, and the open circle the perturbed value. The outlier pulls the average of the labels upward: the learned model parameter shifts by $\widetilde{w} - \widehat{w} = 6/3 = 2$, from the clean solution $\widehat{w} = 3$ (dashed line) to the perturbed solution $\widetilde{w} = 5$ (dotted line)

The effect of perturbations can also be studied on the level of a specific optimization method. For example, the update of online GD becomes

$$\begin{aligned}\mathbf{w}^{(t+1)} &= \mathbf{w}^{(t)} + 2\eta\big(\widetilde{y}^{(t)} - \big(\mathbf{w}^{(t)}\big)^\top\widetilde{\mathbf{x}}^{(t)}\big)\widetilde{\mathbf{x}}^{(t)} \\ &= \mathbf{w}^{(t)} + 2\eta\big(y^{(t)} - \big(\mathbf{w}^{(t)}\big)^\top\mathbf{x}^{(t)}\big)\mathbf{x}^{(t)} + \boldsymbol{\varepsilon}^{(t)}.\end{aligned}$$

Here, $\boldsymbol{\varepsilon}^{(t)}$ is a perturbation term that depends on the data point perturbation $\big(\Delta\mathbf{x}^{(t)}, \Delta y^{(t)}\big)$ and the current model parameters $\mathbf{w}^{(t)}$.

Cross References

regression, linear model, least squares, ERM, squared error loss, GD, online GD, online algorithm, ridge regression, Lasso, pseudoinverse, data point, feature, label, overfitting, Bayes estimator, risk, LMS, SGD

References

1. Bertsekas (2016). Nonlinear Programming. 3rd ed., Athena Scientific, Belmont, MA, USA.

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

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

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

5. Pedregosa et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12(85), pp. 2825–2830.

6. Lehmann and Casella (1998). Theory of Point Estimation. 2nd ed., Springer-Verlag, New York, NY, USA.

7. Jung (2022). Machine Learning: The Basics. Springer Nature, Singapore, Singapore.

8. Paszke et al. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: Adv. Neural Inf. Process. Syst., pp. 8026–8037. Curran Associates Inc.

9. Strang (2016). Introduction to Linear Algebra. 5th ed., Wellesley-Cambridge Press, Wellesley, MA, USA.

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

11. Nesterov (2004). Introductory Lectures on Convex Optimization: A Basic Course. Kluwer Academic, Boston, MA, USA.

12. Papoulis and Pillai (2002). Probability, Random Variables, and Stochastic Processes. 4th ed., McGraw-Hill Higher Education, New York, NY, USA.