Introduction to Machine Learning 2019

Tutorial 2

Harun Mustafa
harun.mustafa@inf.ethz.ch

D-INFK

1 Notation

Vectors are indicated by bold lower-case symbols $\mathbf{x}$, while matrices are indicated by bold upper-case symbols X. The notation $\mathbf{x}^j$ is used to denote the j^{th} component of $\mathbf{x}$. $\mathbf{x}_i$ denotes the i^{th} data point, so $\mathbf{x}_i^j$ denotes the j^{th} component of the i^{th} data point.

2 Linear regression

2.1 Model

Linear regression is a simple linear model which is often used in practice as a baseline model to which more complex models are compared. Given training data $(\mathbf{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$, the response variable y is modeled as

$$y = \mathbf{w}^\top \mathbf{x} + w_0 + \varepsilon, \tag{1}$$

where $\mathbf{w} \in \mathbb{R}^d$ and ε is a random variable which accounts for variation/noise in the measured value of y.

For simplicity of notation, one can transform $\mathbf{w}$ and $\mathbf{x}$ into *homogeneous coordinates* as follows:

$$\tilde{\mathbf{w}} = \begin{bmatrix} \mathbf{w} \\ w_0 \end{bmatrix} \qquad \tilde{\mathbf{x}} = \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix}. \tag{2}$$

Equation 1 can then we rewritten as

$$y = \tilde{\mathbf{w}}^\top \tilde{\mathbf{x}} + \varepsilon. \tag{3}$$

Let us assume, without loss of generality, that in the rest of these notes, the training points $\mathbf{x}_i$ in $\mathbb{R}^d$ have already been transformed in this fashion and originate from data points in $\mathbb{R}^{d-1}$ (i.e., assume that $\mathbf{x}_i^d = 1$ for each $\mathbf{x}_i \in \mathbb{R}^d$).

Given n data points $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \subset \mathbb{R}^d$ with a response variables $\{y_0, \ldots, y_n\} \subset \mathbb{R}$, the training data can be mapped onto a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ and a response vector $\mathbf{y} \in \mathbb{R}^n$, where

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^\top \\ \vdots \\ \mathbf{x}_n^\top \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} y_0 \\ \vdots \\ y_n \end{bmatrix} \tag{4}$$

(i.e., the column vectors $\mathbf{x}_i$ become the rows of $\mathbf{X}$). The model in Equation 3 can then be written jointly for all data as

$$\mathbf{y} = \mathbf{X}\mathbf{w} + \boldsymbol{\varepsilon} \tag{5}$$

(note that $\boldsymbol{\varepsilon}^\top = \begin{bmatrix} \varepsilon_1 \cdots \varepsilon_n \end{bmatrix}$).

2.2 Training by least-squares

Suppose we have a trained weight vector $\mathbf{w}$. We can define the *residues* with respect to $(\mathbf{X}, \mathbf{y})$ as

$$\mathbf{r} = \mathbf{y} - \mathbf{X}\mathbf{w}. \tag{6}$$

We can then define the *least squares* loss as

$$\hat{R}(\mathbf{w}) = \|\mathbf{r}\|_2^2 \tag{7}$$

which for linear regression becomes

$$\begin{aligned}
\hat{R}(\mathbf{w}) &= \|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 \\
&= (\mathbf{y} - \mathbf{X}\mathbf{w})^\top (\mathbf{y} - \mathbf{X}\mathbf{w}) \\
&= \mathbf{y}^\top\mathbf{y} - \mathbf{y}^\top\mathbf{X}\mathbf{w} - \mathbf{w}^\top\mathbf{X}^\top\mathbf{y} + \mathbf{w}^\top\mathbf{X}^\top\mathbf{X}\mathbf{w} \\
&= \mathbf{y}^\top\mathbf{y} - 2\mathbf{y}^\top\mathbf{X}\mathbf{w} + \mathbf{w}^\top\mathbf{X}^\top\mathbf{X}\mathbf{w} && \text{since } \mathbf{y}^\top\mathbf{X}\mathbf{w} = \mathbf{w}^\top\mathbf{X}^\top\mathbf{y} \in \mathbb{R}
\end{aligned}$$

To find the optimal solution $\mathbf{w}^*$, we then solve

$$\begin{aligned}
\mathbf{0} &:= \nabla \hat{R}(\mathbf{w}) \\
&= 2\mathbf{w}^{*\top}\mathbf{X}^\top\mathbf{X} - 2\mathbf{y}^\top\mathbf{X} && \text{since } \frac{\partial \mathbf{x}^T\mathbf{A}\mathbf{x}}{\partial \mathbf{x}} = \mathbf{x}^T(\mathbf{A} + \mathbf{A}^T) \text{ and } (\mathbf{X}^\top\mathbf{X})^\top = \mathbf{X}^\top\mathbf{X}
\end{aligned}$$

which, if $\mathbf{X}^\top\mathbf{X}$ is invertible, can be rearranged to get the closed-form solution

$$\mathbf{w}^* = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y} \tag{8}$$

2.3 Ridge regression and L^p norms

This optimization problem can be made strongly convex by the addition of a *regularization* term to impose restrictions on the complexity of the final trained model

$$\hat{R}_{\text{ridge}}(\mathbf{w}) = \|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_2^2. \tag{9}$$

As an effect of this, the model is less able to fit noise in the training data (see Python demo). Note that this term should only be included during training and not when evaluating error on an independent data set.

2.3.1 L^p norms

This norm is from the family of L^p norms, defined as

$$\|\mathbf{w}\|_p = \sqrt[p]{\sum_{i=1}^{d} |\mathbf{w}^i|^p}. \tag{10}$$

For $p = \infty$, we have the definition

$$\|\mathbf{w}\|_\infty = \max_{i=1}^{d} |\mathbf{w}^i|. \tag{11}$$

Computing its partial derivatives, we get

$$\frac{\partial \|\mathbf{w}\|_p^p}{\partial \mathbf{w}^j} = p|\mathbf{w}^j|^{p-1} sign(\mathbf{w}^j) \tag{12}$$

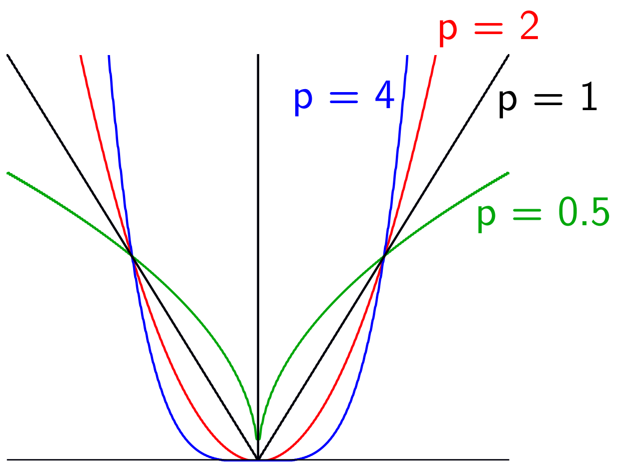

Fig. 1. Plots of the function $y = |x|^p$ for different values of p.

If $p = 1$, then

$$\frac{\partial \|\mathbf{w}\|_1}{\partial \mathbf{w}^j} = sign(\mathbf{w}^j), \tag{13}$$

whereas when $p > 1$, $\frac{\partial \|\mathbf{w}\|_1}{\partial \mathbf{w}^j} \to 0$ as $|\mathbf{w}^j| \to 0$. So when $p > 1$, small non-zero values of $0 < |\mathbf{w}^j| < 1$ tend to approach, but not reach zero due to the decreasing partial derivative. When $p = 1$, this does not happen, and thus, values are more likely to reach zero.

2.4 Interpretability

An interesting property of linear regression is that components $\mathbf{w}^j$ of the weight vector $\mathbf{w}$ can provide a measure of the *importances* of the features of $\mathbf{x}_i$ (their components $\mathbf{x}_i^j$, $j \in \{1, \dots, d\}$). In particular, a large $|\mathbf{w}^j|$ indicates that each $\mathbf{x}_i^j$ has a strong contribution to the solution, and vice versa, and may be considered important.

In general, however, a small value of $|\mathbf{w}_j|$ can also be achieved when a many features are correlated (only one of them contributes to the solution, and the rest are ignored).

2.5 Feature transformations

Non-linear relations between $\mathbf{X}$ and $\mathbf{y}$ may be learned with linear regression by applying a *feature map* $\phi : \mathbb{R}^{d'} \to \mathbb{R}^d$ to the $\mathbf{w}_i$ data prior to training. Then, given data $\mathbf{z}_i \in \mathbb{R}^{d'}$, the model can be represented as

$$y_i = \mathbf{w}^\top \phi(\mathbf{z}_i) \tag{14}$$

Usually, feature maps can be applied if there is some knowledge of the distributions or structure of the $\mathbf{x}_i^j$. For example, given a periodic function

$$y(t) = a_0 + \sum_{n=1}^{\infty} (a_n \cos(nt) + b_n \sin(nt)) + \varepsilon \tag{15}$$

a feature map

$$\phi(t) = [1, \cos(t), \sin(t), \cos(2t), \sin(2t), \dots] \tag{16}$$

may be defined (see Python demo).

3 Gradient descent

In cases where a closed-form solution of $\nabla \hat{R}(\mathbf{w}) = 0$

- cannot be computed
- is computationally expensive to compute
- is ill-conditioned (i.e., small changes in $\mathbf{X}$ lead to large changes in $\mathbf{w}$

gradient descent can be used to compute a local optimum. It is an iterative meta-algorithm structured as follows

1. Initialize $\mathbf{w}_0$ (e.g., randomly, random small values, etc.)
2. Iteratively update
$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla \hat{R}(\mathbf{w}_t) \tag{17}$$
3. Terminate when, for some choice of ε,
$$R(\mathbf{w}_t) - R(\mathbf{w}_{t+1}) \leq \varepsilon \tag{18}$$

where specific algorithms differ in their methods for picking η_t at each step.

3.1 Step size

For sufficiently small η_t, the algorithm converges to some local optimum of $\hat{R}$. In the case of linear regression, this convergence is happens in a linear number of steps for $\eta_t = 0.5$. Setting small η_t increases the amount of computation time, while large η_t may lead to the algorithm not converging (see Figure 2).

In most cases, however, a variable step size is a good way to achieve convergence in an acceptable number of steps.

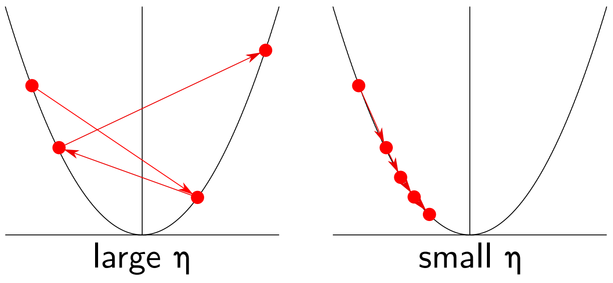

Fig. 2. Effect of learning rate. The choices of vectors $\mathbf{w}_t$ are plotted in red over the function $y = \hat{R}(\mathbf{w})$. Adapted from: A Look at Gradient Descent and RMSprop Optimizers.

3.2 Convexity of $\hat{R}$

When the loss function $\hat{R}$ is convex, we know that all local minima are global,

$$\nabla \hat{R}(\mathbf{w}^*) = 0 \Longrightarrow \hat{R}(\mathbf{w}^*) = \min_{\mathbf{w} \in \mathbb{R}^d} \hat{R}(\mathbf{w}). \tag{19}$$

However, this does not guarantee uniqueness of a particular $\mathbf{w}^*$. In fact, there could be an infinite number of, or no solutions.

For examples, consider the functions

- $f(\mathbf{x}^1, \mathbf{x}^2) = (\mathbf{x}^1)^2$ (convex, infinite number of minimizers)
- $f(x) = x$ (convex, no minimizers)
- $f(x) = e^x$ (strictly convex, no minimizers)