COMP4211

Machine Learning

Larry128

A summary notes for revision

Fall 2024-2025

Contents

1 Linear Regression

1.1 Basic Ideas of Regression

1. Given a training set $S = \{(x^{(l)}, y^{(l)})\}_{l=1}^{N}$ of N labelled examples of input-output pairs.
2. A **Regression Function** $f(\mathbf{x};\mathbf{w})$ uses S such that the predicted output $f(\mathbf{x}^{(\mathbf{l})};\mathbf{w})$ for each input $\mathbf{x}^l$ such that $f(\mathbf{x}^{(l)};\mathbf{w}) \approx \mathbf{y}^l$.
3. (multi-output regression) When the output $\mathbf{y}$ is a vector, it's a multi-output regression.
4. We denote the output by y if the output is univariate.
5. The input $\mathbf{x} = (x_1, \cdots, x_d)^T$ is d-dimensional.

1.2 Linear Regression Function

1. If the regression function is linear, then

$$\begin{aligned} f(\mathbf{x};\mathbf{w}) &= w_0 + w_1 x_1 + \cdots + w_d x_d \\ &= \begin{bmatrix} w_0 & w_1 & \cdots & w_d \end{bmatrix} \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix} & &= \begin{bmatrix} 1 & x_1 & \cdots & x_d \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{bmatrix} \\ &= \mathbf{w}^T \tilde{\mathbf{x}} & &= \tilde{\mathbf{x}}^T \mathbf{w} \end{aligned}$$

2. w_0 is the *bias* term which serves as an offset.
3. The learning problem is to find the best $\mathbf{w}$ according to performance measure on S.

1.3 Loss Function

1. A common way to learn the parameter $\mathbf{w}$ of $f(\mathbf{x};\mathbf{w})$ is to define a loss function $L(\mathbf{w}; S)$
2. The most common loss function is the **squared loss**

$$\begin{aligned} L(\mathbf{w}; S) &= \sum_{l=1}^{N} (f(\mathbf{x}^{(l)};\mathbf{w}) - \mathbf{y}^{(l)})^2 \\ &= \sum_{l=1}^{N} (w_0 + w_1 x_1^{(l)} + \cdots + w_d x_d^{(l)} - y^{(l)})^2 \end{aligned}$$

3. We may also define the loss function by **mean** rather than the sum

$$L(\mathbf{w}; S) = \frac{1}{N} \sum_{l=1}^{N} (f(\mathbf{x}^{(l)};\mathbf{w}) - \mathbf{y}^{(l)})^2$$

4. A special case ($d = 1$)
   Squared loss:

$$L(\mathbf{w}; S) = \sum_{l=1}^{N} (w_0 + w_1 x_1^{(l)} - y^{(l)})^2$$

   We can find the unique optimal solution $\tilde{\mathbf{w}} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}$ that minimizes $L(\mathbf{w}; S)$ using the method of least squares.

First, we take the derivatives of $L(\mathbf{w}; S)$ with respect to w_0 and w_1 and set them to 0.

$$\frac{\partial L}{\partial w_0} = 2\sum_{l=1}^{N}(w_0 + w_1 x_1^{(l)} - y^{(l)}) = 0 \iff \sum_{l=1}^{N}(w_0 + w_1 x_1^{(l)}) = \sum_{l=1}^{N} y^{(l)} \iff Nw_0 + \sum_{l=1}^{N} w_1 x_1^{(l)} = \sum_{l=1}^{N} y^{(l)}$$

$$\frac{\partial L}{\partial w_1} = 2\sum_{l=1}^{N}(w_0 + w_1 x_1^{(l)} - y^{(l)})x_1^{(l)} = 0 \iff w_0 \sum_{l=1}^{N} x_1^{(l)} + w_1 \sum_{l=1}^{N}(x_1^{(l)})^2 = \sum_{l=1}^{N} x_1^{(l)} y^{(l)}$$

Then, we have a system of linear equations of two unknown w_0, w_1. We can write it in matrix form.

$$\mathbf{Aw} = \begin{bmatrix} N & \sum_l x_1^l \\ \sum_l x_1^l & \sum_l (x_1^l)^2 \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} \sum_l y^{(l)} \\ \sum_l x_1^{(l)} y^{(l)} \end{bmatrix} = \mathbf{b}$$

Assuming $\mathbf{A}$ is invertible, the least squares estimate is

$$\tilde{\mathbf{w}} = \mathbf{A}^{-1}\mathbf{b}$$

5. General case ($d \geq 1$)

   (a) (First approach) We express the input and output of N examples as follows

$$\mathbf{X} = \begin{bmatrix} 1 & x_1^l & \cdots & x_d^1 \\ 1 & x_1^2 & \cdots & x_d^2 \\ \vdots & & & \\ 1 & x_1^N & \cdots & x_d^N \end{bmatrix}, \qquad \mathbf{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$$

   Then we can express the matrix form as follows (proof skipped)

$$\mathbf{Aw} = \mathbf{X^T X w} = \mathbf{X^T y} = \mathbf{b}$$

   Therefore, the least squares estimate is

$$\tilde{\mathbf{w}} = (\mathbf{X^T X})^{\mathbf{-1}}\mathbf{X^T y}$$

   , assuming $\mathbf{X^T X}$ is invertible

   (b) (Second approach) First write $\mathbf{Xw} - \mathbf{y}$ as

$$\begin{aligned} \mathbf{Xw} - \mathbf{y} &= \begin{bmatrix} 1 & x_1^l & \cdots & x_d^1 \\ 1 & x_1^2 & \cdots & x_d^2 \\ \vdots & & & \\ 1 & x_1^N & \cdots & x_d^N \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ y_d \end{bmatrix} - \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix} \\ &= \begin{bmatrix} w_0 + w_1 x_1^{(1)} + \cdots + w_d x_d^1 - y^{(1)} \\ w_0 + w_1 x_1^{(2)} + \cdots + w_d x_d^2 - y^{(2)} \\ \vdots \\ w_0 + w_1 x_1^{(N)} + \cdots + w_d x_d^N - y^{(N)} \end{bmatrix} \end{aligned}$$

   Then the squared loss is just the square of **L-2 norm** of $\mathbf{Xw} - \mathbf{y}$

$$L(\mathbf{w}; S) = ||\mathbf{Xw} - \mathbf{y}||^2$$

   We can further write the squared loss as

$$\begin{aligned} L(\mathbf{w}; S) &= ||\mathbf{Xw} - \mathbf{y}||^2 \\ &= (\mathbf{Xw} - \mathbf{y})^T(\mathbf{Xw} - \mathbf{y}) \\ &= (\mathbf{w^T X^T} - \mathbf{y^T})(\mathbf{Xw} - \mathbf{y}) \\ &= \mathbf{w^T X^T X w} - \mathbf{2y^T X w} + \mathbf{y^T y} \end{aligned}$$

After that, we can take the derivative with respect to $\mathbf{w}$

$$\begin{aligned}
&\frac{\partial L}{\partial \mathbf{w}} = 2\mathbf{X^T X w} - 2\mathbf{X^T y} = 0 \\
\Longleftrightarrow\ &\mathbf{X^T X w} = \mathbf{X^T y} \\
\Longleftrightarrow\ &\tilde{\mathbf{w}} = (\mathbf{X^T X})^{\mathbf{-1}}\mathbf{X^T y}
\end{aligned}$$

(c) Complexity considerations
To compute $\tilde{\mathbf{w}}$, we need to invert $\mathbf{X^T X} \in \mathbb{R}^{(d+1)\times(d+1)}$. *LeGall* is the fastest algorithm to compute that with $O(n^{2.3728639})$, instead of $O(n^3)$ for Cholesky, LU, Gaussian elimination.