Logistic dan Softmax Regression: Formulasi, Optimisasi, dan Implementasi

1. Formulasi Model

1.1. Klasifikasi Biner (Logistic Regression)

Model memetakan fitur $\mathbf{x} \in \mathrm{R}^n$ ke probabilitas kelas positif melalui fungsi sigmoid:

$$p(y{=}1 \mid \mathbf{x}) = \sigma(\mathbf{x}^\top \mathbf{w} + b) = \frac{1}{1 + \exp(-(\mathbf{x}^\top \mathbf{w} + b))}.$$

Parameter yang digunakan mencakup *learning rate*, jumlah iterasi maksimum, jenis dan kekuatan regularisasi, pilihan *optimizer* (*gradient descent* atau *Newton*), serta *tolerance* untuk kriteria henti.

1.2. Klasifikasi Multikelas (Softmax Regression)

Untuk K kelas, probabilitas dihitung oleh *softmax*:

$$p(y{=}k \mid \mathbf{x}) = \frac{\exp(\mathbf{x}^\top \mathbf{W}_{:,k} + b_k)}{\sum_{j=1}^{K} \exp(\mathbf{x}^\top \mathbf{W}_{:,j} + b_j)}.$$

Implementasi menggunakan *softmax* yang distabilkan numerik dengan mengurangkan $\max$ pada dimensi logit.

2. Fungsi Kerugian dan Regularisasi

2.1. Biner (Binary Cross-Entropy)

$$\mathrm{L}_{\text{bin}} = -\frac{1}{m} \sum_{i=1}^{m} [y_i \log p_i + (1 - y_i) \log(1 - p_i)] \; + \; \lambda \, \Omega(\mathbf{w}),$$

dengan $\Omega(\mathbf{w}) = \|\mathbf{w}\|_1$ (L1) atau $\|\mathbf{w}\|_2^2$ (L2). Gradien menambahkan $\lambda \, \mathrm{sign}(\mathbf{w})$ untuk L1 atau $2\lambda\mathbf{w}$ untuk L2.

2.2. Multikelas (Cross-Entropy)

$$\mathrm{L}_{\text{mc}} = -\frac{1}{m} \sum_{i=1}^{m} \sum_{k=1}^{K} y_{ik} \log p_{ik} \; + \; \lambda \, \Omega(\mathbf{W}).$$

Probabilitas *softmax* diklip ke $[10^{-15}, 1 - 10^{-15}]$ sebelum log untuk stabilitas; penalti L1/L2 ditambahkan pada *cost*.

3. Optimisasi

3.1. *Gradient Descent*

Biner.

$$\nabla_{\mathbf{w}} = \frac{1}{m}\mathbf{X}^{\top}(\mathbf{p} - \mathbf{y}) + \text{reg}, \quad \nabla_b = \frac{1}{m}\sum_i (p_i - y_i),$$

dengan pembaruan $\mathbf{w} \leftarrow \mathbf{w} - \eta\nabla_{\mathbf{w}},\ b \leftarrow b - \eta\nabla_b$.

Multikelas.

$$\nabla_{\mathbf{W}} = \frac{1}{m}\mathbf{X}^{\top}(\mathbf{P} - \mathbf{Y}) + \text{reg}, \quad \nabla_{\mathbf{b}} = \frac{1}{m}\sum_i (\mathbf{P}_i - \mathbf{Y}_i),$$

pembaruan $\mathbf{W} \leftarrow \mathbf{W} - \eta\nabla_{\mathbf{W}},\ \mathbf{b} \leftarrow \mathbf{b} - \eta\nabla_{\mathbf{b}}$.

3.2. *Newton's Method* (khusus biner)

Hessian dihitung sebagai

$$\mathbf{H} = \frac{1}{m}\mathbf{X}^{\top}\text{Diag}(\mathbf{p} \odot (1 - \mathbf{p}))\mathbf{X} \ + \ 2\lambda\mathbf{I} \quad \text{(bila L2)}.$$

Langkah Newton: $\mathbf{w} \leftarrow \mathbf{w} - \mathbf{H}^{-1}\nabla_{\mathbf{w}}$; bias diperbarui via gradien. Jika inversi gagal, dilakukan *fallback* ke *gradient descent*.

3.3. Kriteria Henti

Pelatihan berhenti dini bila selisih biaya berturut-turut di bawah *tolerance*, memanfaatkan *cost history*.

4. Detil Implementasi

Bobot diinisialisasi kecil acak; bias nol. Untuk multikelas, label di-*one-hot* sebelum optimisasi; *softmax* memakai stabilisasi numerik; probabilitas keluaran diklip sebelum perhitungan *cross-entropy*.

5. Prediksi

Untuk kasus klasifikasi dua kelas, probabilitas dihitung menggunakan fungsi sigmoid:

$$\text{predict\_proba}(X) = \sigma(X\mathbf{w} + b)$$

Prediksi kelas akhir dilakukan berdasarkan ambang batas 0.5:

$$\text{predict}(X) = 1\,[\text{predict\_proba}(X) \geq 0.5]$$