XGBNPClassifier, XGBFairClassifier, LGBMNPClassifier and LGBMFairClassifier Manual

xxx

April 2, 2023

1 XGBNPClassifier and LGBMNPClassifier

1.1 Binary Classification Neyman Pearson Problem

Consider Neyman Pearson Problem with surrogate loss function in binary classification

$$\begin{aligned} \min_{f_i, \forall i \in [n]} \quad & \sum_i \ell(f_i, y_i = 1) \\ \text{s.t.} \quad & \sum_i \ell(f_i, y_i = 0) \leq \alpha, \end{aligned} \tag{1}$$

where $\ell(f_i, y_i) = -y_i \log(\frac{1}{1+\exp(-f_i)}) - (1-y_i)\log(1 - \frac{1}{1+\exp(-f_i)})$ and we denote $\hat{y}_i = \frac{1}{1+\exp(-f_i)}$ in binary classification. Now, we apply Accelerated Primal Dual Algorithm (APD) [1] to solve this convex constraint optimization, which is stated as Algorithm 1.

Now, we aim to use xgboost to help us to solve the sub-problem, which is based on following gradient and hessian. Denote the $\mathcal{L}(\mathbf{y}, f, \lambda) = \sum_i \ell(f_i, y_i = 1) + \lambda(\sum_i \ell(f_i, y_i = 0) - \alpha), \hat{y}_i = \frac{1}{1+\exp(-f_i)}$, $\partial_{f_i} \hat{y}_i = \hat{y}_i(1-\hat{y}_i)$, then we have

$$\begin{aligned} g_i &= \partial_{f_i} \mathcal{L}(\mathbf{y}, f, \lambda^t) + \tfrac{1}{2\tau^t}\|f - f^t\|^2 \\ &= -y_i(1-\hat{y}_i) + \lambda^t(1-y_i)\hat{y}_i + \tfrac{1}{\tau^t}(f_i - f_i^t) \\ &= -y_i + (y_i + \lambda^t(1-y_i))\hat{y}_i + \tfrac{1}{\tau^t}(f_i - f_i^t) \end{aligned} \tag{2}$$

$$\begin{aligned} h_i &= y_i(\hat{y}_i - \hat{y}_i^2) + \lambda^t(1-y_i)(\hat{y}_i - \hat{y}_i^2) + \tfrac{1}{\tau^t} \\ &= (y_i + \lambda^t(1-y_i))(\hat{y}_i - \hat{y}_i^2) + \tfrac{1}{\tau^t} \end{aligned} \tag{3}$$

1.2 Neyman Pearson Binary Classification with 1 and −1 label (LightGBM)

$$\begin{aligned} \min_{f_i, \forall i \in [n]} \quad & \sum_i \ell(f_i, y_i = 1) \\ \text{s.t.} \quad & \sum_i \ell(f_i, y_i = -1) \leq \alpha, \end{aligned} \tag{4}$$

Algorithm 1 APD for Binary Neyman Pearson Classification

Require: $f^{-1} = f^0, \lambda^{-1} = \lambda^0, \tau^0, \sigma^0, \theta^t = 1$.

1: **for** $t = 0, 1, \ldots, T-1$ **do**
2: $z^t = \sum_i^n \ell(f_i^t, y_i = 0) - \alpha + \theta^t(\sum_i^n \ell(f_i^t, y_i = 0) - \sum_i^n \ell(f_i^{t-1}, y_i = 0))$
3: $\lambda^{t+1} = [\lambda^t + \sigma^t z^t]_+$
4: $f^{t+1} = \text{argmin}_f \sum_i \ell(f_i, y_i = 1) + (\lambda^{t+1})(\sum_i \ell(f_i, y_i = 0)) + \frac{1}{2\tau^t}\|f - f^t\|^2$
5: **end for**
6: **Output:** f^T

Algorithm 2 Inexact Accelerated Primal Dual Algorithm (APD) for Multi-class Neyman Pearson Classification

Require: $f^{-1} = f^0, \lambda^{-1} = \lambda^0$, $\tau^0, \sigma^0, \theta^t = 1$.

1: **for** $t = 0, 1, \ldots, T-1$ **do**

2: $z_k^t = \sum_{i=1}^n \ell(y_{ik}, f_{ik}^t) - \alpha_k + \theta^t(\sum_{i=1}^n \ell(y_{ik}, f_{ik}^t) - \sum_{i=1}^n \ell(y_{ik}, f_{ik}^{t-1}))$

3: $\lambda_k^{t+1} = [\lambda_k^t + \sigma^t z_k^t]_+$

4: $f^{t+1} = \text{argmin}_f \sum_{i=1}^n \sum_{k=1}^K (w_k + \lambda_k^{t+1}) \ell(y_{ik}, f_{ik}) + \frac{1}{2\tau^t}\|f - f^t\|^2$

5: **end for**

6: **Output:** f^T

where $\ell(f_i, y_i) = \log(1 + \exp(-y_i f_i))$. Note that sigmoid function $\hat{y}_i = \frac{1}{1+\exp(-y_i f_i)}$, $\ell(\mathbf{y}, f, \lambda) = \sum_i \ell(f_i, y_i = 1) + \lambda(\sum_i \ell(f_i, y_i = -1))$, then the gradient is $\partial_{f_i}\ell(f_i, y_i) = -y_i(1 - \hat{y}_i) = -y_i \cdot \frac{1}{1+\exp(y_i f_i)}$, and $\partial_{f_i}^2 \ell(f_i, y_i) = y_i^2(1 - \hat{y}_i)\hat{y}_i = (1 - \hat{y}_i)\hat{y}_i$. Hence the gradient and hessian of LGBMNPClassifier needs are calculated as following:

$$\begin{aligned} g_i &= -y_i \cdot \tfrac{1}{1+\exp(y_i \cdot f_i)} \cdot \mathbf{1}(y_i = 1) - \lambda \cdot y_i \cdot \tfrac{1}{1+\exp(y_i \cdot f_i)} \mathbf{1}(y_i = -1) + \tfrac{1}{\tau}(f_i - f_i^t) \\ &= (-\mathbf{1}(y_i = 1) - \lambda \cdot \mathbf{1}(y_i = -1)) \cdot y_i \cdot \tfrac{1}{1+\exp(y_i \cdot f_i)} + \tfrac{1}{\tau}(f - f_i^t) \end{aligned} \tag{5}$$

$$h_i = (\mathbf{1}(y_i = 1) + \lambda \cdot \mathbf{1}(y_i = -1)) \cdot (1 - \hat{y}_i)\hat{y}_i + \tfrac{1}{\tau} \tag{6}$$

1.3 Multi-class Neyman Pearson Classification

$$\begin{aligned} \min_{f_{ik}, \forall i \in [n], k \in [K]} \quad & \sum_{i=1}^n \sum_{k=1}^K w_k \ell(f_{ik}, y_{ik}) \\ \text{s.t.} \quad & \sum_{i=1}^n \ell(f_{ik}, y_{ik}) \le \alpha_k, \forall k \in [K], \end{aligned} \tag{7}$$

where w_k is pre-specific parameters for specific classes, $\ell(y_{ik}, f_{ik}) = -y_{ik} \log \frac{\exp(f_{ik})}{\sum_{k=1}^K \exp(f_{ik})} = -y_{ik} \log \hat{y}_{ik}$.

Now, we aim to use xgboost to help us to solve the sub-problem, which is based on following gradient and hessian. Denote the $\mathcal{L}(\mathbf{y}, f, \lambda) = \sum_{i=1}^n \sum_{k=1}^K (w_k + \lambda_k)\ell(y_{ik}, f_{ik}), \hat{y}_{ik} = \frac{\exp(f_{ik})}{\sum_{k=1}^K \exp(f_{ik})}$, then

$$\begin{aligned} g_{ij} &= \sum_{k=1, k\neq j}^K (w_k + \lambda_k) y_{ik} \hat{y}_{ij} + \tfrac{1}{\tau^t}(f_{ij} - f_{ij}^t) + (w_j + \lambda_j) y_{ij}(\hat{y}_{ij} - 1) \\ &= [\hat{y}_{ij} \sum_{k=1}^K (w_k + \lambda_k) y_{ik}] - (w_j + \lambda_j) y_{ij} + \tfrac{1}{\tau^t}(f_{ij} - f_{ij}^t), \end{aligned} \tag{8}$$

$$\begin{aligned} h_{ij} &= \sum_{k=1}^K (w_k + \lambda_k) y_{ik} \hat{y}_{ij}(1 - \hat{y}_{ij}) + \tfrac{1}{\tau^t} \\ &= \hat{y}_{ij}(1 - \hat{y}_{ij}) \sum_{k=1}^K (w_k + \lambda_k) y_{ik} + \tfrac{1}{\tau^t}, \end{aligned} \tag{9}$$

where w_k is `class_weight_w`, λ is `lam` and $\lambda + w$ is `class_weight`.

2 XGBFairClassifier and LGBMFairClassifier

Choose LCP as outer algorithm and inner call APD to solve.

Algorithm 3 Inexact Constrained Proximal Point [2] for Binary Classification with Fairness Constraint

Require: $\rho = \frac{1}{4}$, f^0, α_{ab}^l.

1: **for** $l = 0, 1, \ldots, L-1$ **do**

2: Call APD (Algorithm 4) to find $\varepsilon-$ solution f^{l+1} the following problem

$$\begin{aligned} \min_{f} \quad & \sum_{s\in\mathcal{S}} \sum_{i} \ell(f_i, y_i \mid S_i = s) + \tfrac{\rho}{2}\|f - f^l\|^2 \\ \text{s.t.} \quad & \tfrac{\beta}{n_a} \sum_{i} \ell(f_i, y_i | S_i = a) - \tfrac{\beta}{n_b} \sum_{i} \ell(f_i, y_i | S_i = b) + \tfrac{\beta\rho}{2}\|f - f^l\|^2 \le \beta\alpha_{ab}^l, \forall a, b \in \mathcal{S} \end{aligned} \tag{13}$$

3: **end for**

4: **Output:** f^L

Algorithm 4 APD for sub-problem in ICP

Require: $f^{-1} = f^0, \lambda^{-1} = \lambda^0$, $\tau^0, \sigma^0, \theta^0 = 1$.

1: **for** $t = 0, 1, \ldots, T-1$ **do**

2: $e_{ab}^t = (\frac{\beta}{n_a} \sum_i \ell(f_i^t, y_i | S_i = a) - \frac{\beta}{n_b} \sum_i \ell(f_i^t, y_i | S_i = b) + \frac{\beta\rho}{2}\|f^t - f^l\|^2) - (\frac{\beta}{n_a} \sum_i \ell(f_i^{t-1}, y_i | S_i = a) - \frac{\beta}{n_b} \sum_i \ell(f_i^{t-1}, y_i | S_i = b) + \frac{\beta\rho}{2}\|f^{t-1} - f^l\|^2)$

3: $z_{ab}^t = \frac{\beta}{n_a} \sum_i \ell(f_i^t, y_i | S_i = a) - \frac{\beta}{n_b} \sum_i \ell(f_i^t, y_i | S_i = b) + \frac{\beta\rho}{2}\|f^t - f^l\|^2 - \beta\alpha_{ab}^l + \theta^t e_{ab}^t$

4: $\lambda_{ab}^{t+1} = [\lambda_{ab}^t + \sigma^t z_{ab}^t]_+$

5: $f^{t+1} = \text{argmin}_f$ (12)

6: **end for**

7: **Output:** f^T

2.1 Binary Classification Fairness Problem

$$\begin{aligned} \min_{f_i, \forall i\in[n]} \quad & \sum_i \ell(f_i, y_i) = \sum_{j\in\mathcal{S}} \sum_i \ell(f_i, y_i \mid S_i = j) \\ \text{s.t.} \quad & \tfrac{1}{n_a} \sum_i \ell(f_i, y_i | S_i = a) - \tfrac{1}{n_b} \sum_i \ell(f_i, y_i | S_i = b) \le \alpha_{ab}, \forall a, b \in \mathcal{S}, a \ne b, \end{aligned} \tag{10}$$

where $\mathcal{S}$ is sensitive attribute, $n_j = |D_{S=j}|$ is the size of dataset with j attribute and $\sum_i \ell(f_i, y_i) = -y_i \log(\frac{1}{1+\exp(-f_i)}) - w_0(1-y_i)\log(1 - \frac{1}{1+\exp(-f_i)})$. To make the objective and constraint is more balance, then we multiply `scale_factor`(β) for every constraint. Hence we apply the following Algorithm 3 to solve (10). For constraint $\frac{\beta}{n_a} \sum_i \ell(f_i, y_i | S_i = a) - \frac{\beta}{n_b} \sum_i \ell(f_i, y_i | S_i = b) \le \beta\alpha_{ab}^l$, we give the corresponding dual variable λ_{ab}. Then the dual variable can be write as a matrix Λ:

$$\Lambda := \begin{bmatrix} 0 & \lambda_{12} & \lambda_{13} & \cdots & \lambda_{1|\mathcal{S}|} \\ \lambda_{21} & 0 & \lambda_{23} & \cdots & \lambda_{2|\mathcal{S}|} \\ \lambda_{31} & \lambda_{32} & 0 & \cdots & \lambda_{3|\mathcal{S}|} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \lambda_{|\mathcal{S}|1} & \lambda_{|\mathcal{S}|2} & \lambda_{|\mathcal{S}|3} & \cdots & 0 \end{bmatrix} \tag{11}$$

The key sub-problem for GBM can be written as following:

$$\begin{aligned} & \sum_{a\in\mathcal{S}} (\tfrac{\beta\cdot(\sum_{j\in\mathcal{S}, j\ne a} \lambda_{aj} - \sum_{j\in\mathcal{S}, j\ne a} \lambda_{ja})}{n_a} + 1) \sum_i \ell(f_i, y_i \mid S_i = a) \\ & + (\tfrac{\rho}{2} + \tfrac{\beta\rho}{2} \sum_{b,c\in\mathcal{S}, c\ne b} \lambda_{bc}) \|f - f^l\|^2 + \tfrac{1}{2\tau^t}\|f - f^t\|^2. \end{aligned} \tag{12}$$

The gradient and hessian of (12) for $\sum_i \ell(f_i, y_i) = \sum_i (-y_i \log(\frac{1}{1+\exp(-f_i)}) - w_0(1-y_i)\log(1 -$

$\frac{1}{1+\exp(-f_i)}$)) are given as follows:

$$\begin{aligned} g_i =&(\frac{\beta\cdot(\sum_{j\in\mathcal{S},j\neq a}\lambda_{aj}-\sum_{j\in\mathcal{S},j\neq a}\lambda_{ja})}{n_a}+1)(-y_i+(y_i+w_0(1-y_i))\hat{y}_i) \\ &+(\rho+\beta\rho\sum_{b,c\in\mathcal{S},c\neq b}\lambda_{bc})(f-f^l)+\frac{1}{\tau^t}(f-f^t), \qquad , s_i=a, \end{aligned} \tag{14}$$

and

$$\begin{aligned} h_i =&(\frac{\beta\cdot(\sum_{j\in\mathcal{S},j\neq a}\lambda_{aj}-\sum_{j\in\mathcal{S},j\neq a}\lambda_{ja})}{n_a}+1)(y_i+w_0(1-y_i))(\hat{y}_i-\hat{y}_i^2) \\ &+(\rho+\beta\rho\sum_{b,c\in\mathcal{S},c\neq b}\lambda_{bc})+\frac{1}{\tau^t}, \qquad , s_i=a. \end{aligned} \tag{15}$$

where $\frac{\beta\cdot(\sum_{j\in\mathcal{S},j\neq a}\lambda_{aj}-\sum_{j\in\mathcal{S},j\neq a}\lambda_{ja})}{n_a}+1$ is parameter `attribute_weight1`, $(\rho+\beta\rho\sum_{b,c\in\mathcal{S},c\neq b}\lambda_{bc})$ is parameter `attribute_weight2`.

2.2 Fairness Binary Classification with label -1 and 1

The gradient and hessian are given as follows:

$$\begin{aligned} g_i =&(\frac{\beta\cdot(\sum_{j\in\mathcal{S},j\neq a}\lambda_{aj}-\sum_{j\in\mathcal{S},j\neq a}\lambda_{ja})}{n_a}+1)\cdot(-y_i\cdot\frac{1}{1+\exp(y_if_i)}) \\ &+(\rho+\beta\rho\sum_{b,c\in\mathcal{S},c\neq b}\lambda_{bc})(f-f^l)+\frac{1}{\tau^t}(f-f^t), \end{aligned} \tag{16}$$

and

$$\begin{aligned} h_i =&(\frac{\beta\cdot(\sum_{j\in\mathcal{S},j\neq a}\lambda_{aj}-\sum_{j\in\mathcal{S},j\neq a}\lambda_{ja})}{n_a}+1)\cdot(\hat{y}_i(1-\hat{y}_i)) \\ &+(\rho+\beta\rho\sum_{b,c\in\mathcal{S},c\neq b}\lambda_{bc})+\frac{1}{\tau^t}. \end{aligned} \tag{17}$$

2.3 Muti-class Fairness Problem

$$\begin{aligned} \min_{f_i,\forall i\in[n]} \quad & \sum_i \ell(f_i,y_i)=\sum_{j\in\mathcal{S}}\sum_i \ell(f_i,y_i\mid S_i=j) \\ \text{s. t.} \quad & \frac{1}{n_a}\sum_i \ell(f_i,y_i|S_i=a)-\frac{1}{n_b}\sum_i \ell(f_i,y_i|S_i=b)\leq\alpha_{ab},\forall a,b\in\mathcal{S},a\neq b \end{aligned} \tag{18}$$

where $\mathcal{S}$ is sensitive attribute set, $n_j=|D_{S=j}|$ is the size of dataset with j attribute and $\sum_i \ell(f_i,y_i)=-\sum_{i=1}^n\sum_{k=1}^K w_k y_{ik}\log(\frac{\exp(f_{ik})}{\sum_{k=1}^K\exp(f_{ik})})$. Then the gradient and hessian in the context can be shown as

$$\begin{aligned} g_{ij} =&(\frac{\beta\cdot(\sum_{j\in\mathcal{S},j\neq a}\lambda_{aj}-\sum_{j\in\mathcal{S},j\neq b}\lambda_{jb})}{n_a}+1)((\hat{y}_{ij}\sum_{k=1}^K w_k y_{ik})-w_j y_{ij}) \\ &+(\rho+\beta\rho\sum_{b,c\in\mathcal{S},c\neq b}\lambda_{bc})(f-f^l)+\frac{1}{\tau^t}(f-f^t), \qquad , s_i=a, \end{aligned} \tag{19}$$

and

$$\begin{aligned} h_{ij} =&(\frac{\beta\cdot(\sum_{j\in\mathcal{S},j\neq a}\lambda_{aj}-\sum_{j\in\mathcal{S},j\neq b}\lambda_{jb})}{n_a}+1)(\hat{y}_{ij}(1-\hat{y}_{ij})\sum_{k=1}^K w_k y_{ik}) \\ &+(\rho+\beta\rho\sum_{b,c\in\mathcal{S},c\neq b}\lambda_{bc})+\frac{1}{\tau^t}, \qquad , s_i=a, \end{aligned} \tag{20}$$

where $\frac{\beta\cdot(\sum_{j\in\mathcal{S},j\neq a}\lambda_{aj}-\sum_{j\in\mathcal{S},j\neq a}\lambda_{ja})}{n_a}+1$ is parameter `attribute_weight1`, $(\rho+\beta\rho\sum_{b,c\in\mathcal{S},c\neq b}\lambda_{bc})$ is parameter `attribute_weight2`.

3 How to terminate algorithms

3.1 How to terminate inner loop APD in IQRC with heuristic method?

In theory, our sub-problem must require the solution to be sufficiently exact. However, this needs the optimal values to determine the "exact". Here we give a heuristic method to terminate algorithms that has been found to be very effective in practice (For Algorithm 1, 2, 4).

- Maintain a matrix $\Lambda = [\lambda_{t-4}, \lambda_{t-3}, \lambda_{t-2}, \lambda_{t-1}, \lambda_t]$ to record the last 5 dual variables.
- Take $\lambda_{max} = \max(\Lambda, axis = 1)$, i.e., taking maximum by row.
- Take $\lambda_{min} = \min(\Lambda, axis = 1)$, i.e., taking minimum by row.
- Take $\lambda_{mean} = mean(\Lambda, axis = 1)$, i.e., taking mean by row.
- Terminate algorithm if $\frac{\|(\lambda_{\max} - \lambda_{\min})/2 - \lambda_{mean}\|}{\|\lambda_{mean}\|} < 1e-3$.

4 Outlier detection

It is well known that the cross-entropy loss function as a proxy function for the 0-1 loss function may not respond well to control type II errors and fairness in a realistic way due to the presence of outliers. Therefore, a remedy strategy is to perform outlier detection, i.e., remove the data points with a large loss after a certain number of iterative steps. The strategy can be summaried as following steps:

- Calculate the loss of all data points $\ell_i = -y_i \log \hat{y}_i - (1-y_i)\log(1-\hat{y}_i)$ or $\ell_i = -\sum_{k=1}^{K} y_{ik} \log \hat{y}_{ik}$.
- Sort all the losses and choose the value with smaller 90% loss to calculate the mean value $\bar{\ell}_{0.9}$.
- Filter the data points from the larger 10% with a loss greater than `outlier_threshold` times the average loss ($\bar{\ell}_{0.9}$) to remove, where `outlier_threshold` is a hyper-parameter.
- Additional case: an extreme case is that we happen to remove all the sample points belonging to a certain sensitive attribute s or a certain class k. At this point, the algorithm chooses to increase this hyperparameter `outlier_threshold` ($\times 2$) to reduce the number of deleted samples until there is one sample for the next training set.

5 How to set the upper bound in constraint

How to set α in (1)?

In binary Neyman Pearson Classification, the constraint is $-\sum_i (1-y_i)\log(1-\hat{y}_i) \le \alpha$. Hence the heuristic strategy we first set $\alpha = n_0 \cdot e \cdot (-\log \frac{1}{2})$, where e is expected type II error rate, n_0 is the dataset size of class 0. After outlier detection describe in 4, we change the $\alpha = -n_0 e \frac{\sum_i (1-y_i)\log(1-\hat{y}_i)}{n_0} = -e \sum_i (1-y_i)\log(1-\hat{y}_i)$, where $\hat{y}_i$ is the evaluated value after outlier detection.

How to set α_k in (7)?

Similarly, we first set $\alpha_k = n_k \cdot e_k \cdot (-\log \frac{1}{K}) = n_k e_k \log K$, where e_k is expected error rate of class k and n_k is the dataset size of class k. After outlier detection describe in 4, we change the $\alpha_k = -n_k e_k \frac{\sum_i \sum_{j=1}^{K} y_{ij} \log \hat{y}_{ij} \mathbf{1}(c_i = k)}{n_k} = -e_k (\sum_i \sum_{j=1}^{K} y_{ij} \log \hat{y}_{ij})$, where $\hat{y}_{ij}$ is the evaluated value after outlier detection.

References

[1] E. Y. Hamedani and N. S. Aybat, *A primal-dual algorithm with line search for general convex-concave saddle point problems*, SIAM Journal on Optimization, 31 (2021), pp. 1299–1329.

[2] Z. Jia and B. Grimmer, *First-order methods for nonsmooth nonconvex functional constrained optimization with or without slater points*, arXiv preprint arXiv:2212.00927, (2022).

In the appendix, I will give a detailed derivation of all the formulas. Since these formulas are rather complicated. Therefore, there may be some errors. The derivation process will be put here, and you are welcome to correct the errors.

A Details in Section 1 and Section 2

A.1 Details in Section 1.1

A.1.1 How to derive (2)

Since $\mathcal{L}(\mathbf{y}, f, \lambda) = -\sum_i (y_i \log(\hat{y}_i) + \lambda(1-y_i)\log(1-\hat{y}_i)) - \lambda\alpha$, and $\partial_{f_i}\hat{y}_i = \frac{\exp(-f_i)}{(1+\exp(-f_i))^2} = \hat{y}_i(1-\hat{y}_i)$, then we have

$$\begin{aligned}\partial_{f_i}\mathcal{L}(\mathbf{y}, f, \lambda^t) &= -(\tfrac{y_i}{\hat{y}_i}\hat{y}_i(1-\hat{y}_i) + \tfrac{\lambda(1-y_i)}{1-\hat{y}_i}(-\hat{y}_i(1-\hat{y}_i))) \\ &= -y_i(1-\hat{y}_i) + \lambda(1-y_i)\hat{y}_i\end{aligned}$$

A.1.2 How to calculate (3)

It follows from $\partial_{f_i}\hat{y}_i = \hat{y}_i(1-\hat{y}_i)$ that

$$\partial^2_{f_i}\mathcal{L}(\mathbf{y}, f, \lambda^t) = (y_i + \lambda(1-y_i))\partial_{f_i}\hat{y}_i = (y_i + \lambda(1-y_i))\hat{y}_i(1-\hat{y}_i) \tag{21}$$

A.2 Details in Section 1.3

A.2.1 How to derive (8)

Recall the loss function is $\mathcal{L}(\mathbf{y}, f, \lambda) = -\sum_{i=1}^n \sum_{k=1}^K (w_k + \lambda_k) y_{ik} \log \hat{y}_{ik}$, where $\hat{y}_{ik} = \frac{\exp(f_{ik})}{\sum_{k=1}^K \exp(f_{ik})}$, and

$$\partial_{f_{ij}}\hat{y}_{ik} = \begin{cases} \frac{\exp(f_{ij})\sum_{k=1}^K \exp(f_{ik}) - (\exp(f_{ij}))^2}{(\sum_{k=1}^K \exp(f_{ik}))^2} = \hat{y}_{ij} - \hat{y}^2_{ij} & j = k \\ \frac{-\exp(f_{ij})\exp(f_{ik})}{(\sum_{k=1}^K \exp(f_{ik}))^2} = -\hat{y}_{ik}\hat{y}_{ij} & j \neq k \end{cases} \tag{22}$$

$$\partial_{f_{ij}}(-y_{ik}\log\hat{y}_{ik}) = -\tfrac{y_{ik}}{\hat{y}_{ik}}\partial_{f_{ij}}\hat{y}_{ik} = \begin{cases} y_{ij}\hat{y}_{ij} - y_{ij} & j = k \\ y_{ik}\hat{y}_{ij} & j \neq k \end{cases} \tag{23}$$

Hence,

$$\begin{aligned}\partial_{f_{ij}}\mathcal{L}(\mathbf{y}, f, \lambda) &= \sum_{k=1, k\neq j}^K (w_k + \lambda_k) y_{ik}\hat{y}_{ij} + (w_j + \lambda_j)(y_{ij}\hat{y}_{ij} - y_{ij}) \\ &= (\hat{y}_{ij}\sum_{k=1}^K (w_k + \lambda_k) y_{ik}) - (w_j + \lambda_j) y_{ij}.\end{aligned}$$

A.2.2 How to derive (9)

It follows from (22) and (23) that

$$\partial^2_{f_{ij}}\mathcal{L}(\mathbf{y}, f, \lambda) = (\hat{y}_{ij} - \hat{y}^2_{ij})(\sum_{k=1}^K (w_k + \lambda_k) y_{ik}). \tag{24}$$

A.3 Details in Section 2.1 and 2.3

Omit here.