هوش مصنوعی

بهار ۱۴۰۰

استاد: محمدحسین رهبان

گردآورندگان: محمدمهدی اصمع

دانشگاه صنعتی شریف

دانشکده‌ی مهندسی کامپیوتر

پاسخ تمرین ششم، بخش اول | رگرسیون و پرسپترون | مهلت ارسال: ـ

---

۱. (نمره)

(آ)

$$Y \sim Bernoulli(p) \Rightarrow \mathbb{P}(Y) = p^Y (1-p)^{(1-Y)}$$

$$\mathbb{P}(y_1, \cdots, y_N; p) = \prod_{i=1}^{N} \mathbb{P}(y_i; p) = \prod_{i=1}^{N} p^{y_i}(1-p)^{(1-y_i)}$$

$$\log \mathbb{P}(y_1, \cdots, y_N; p) = \sum_{i=1}^{N} \log(p^{y_i}(1-p)^{(1-y_i)}) = \sum_{i=1}^{N} \{y_i \log(p) + (1-y_i)\log(1-p)\}$$

(ب) ابتدا p را با استفاده از رابطه داده شده به دست می‌آوریم:

$$\log(\frac{p}{1-p}) = ax + b \Rightarrow \frac{p}{1-p} = e^{ax+b} \Rightarrow \frac{1}{1-p} = 1 + e^{ax+b}$$

$$p = \frac{e^{ax+b}}{1+e^{ax+b}} = \frac{1}{1+e^{-(ax+b)}} = \sigma(-(ax+b))$$

حال p را در رابطه‌ی قسمت (آ) جایگذاری می‌کنیم:

$$\hat{p} \ = \arg\min_{p} - \sum_{i=1}^{N} \log(p^{y_i}(1-p)^{(1-y_i)}) = \sum_{i=1}^{N} \{y_i \log(p) + (1-y_i)\log(1-p)\}$$

$$\hat{a}, \hat{b} = \arg\min_{a,b} \sum_{i=1}^{N} \log(p^{y_i}(1-p)^{(1-y_i)})$$

$$= \arg\min_{a,b} \sum_{i=1}^{N} \{y_i \log(\sigma(-(ax_i+b))) + (1-y_i)\log(1-\sigma(-(ax_i+b)))\}$$

که این همان تابع loss رگرسیون لجستیک است.

۲. (نمره)

(آ) به جای اینکه $p(C_1|x)$ را تخمین بزنیم، هر $p(C_i|x)$ را به صورت زیر در نظرمی‌گیریم:

$$p(C_i|x) = y_i(x) = \frac{\exp(-w_i^T x)}{\sum_{j=1}^{K} \exp(-w_j^T x)}$$

T را ماتریسی $N \times K$ در نظر می‌گیریم که هر سطر آن بردار one-hot انکود شده‌ی $t^{(i)}$ است و هر دیتاپوینت را به شکل $x^{(i)}, t^{(i)}$ در نظر می‌گیریم. تابع loss برای این مدل به صورت زیر است:

$$p(t^{(i)}|x^{(i)}, w) = y_{t^{(i)}}(x^{(i)})$$

$$L(w) = -\sum_{n=1}^{N}\{\sum_{k=1}^{K} T_{nk} \log(y_k(x^{(n)}))\}$$

(ب) کد پایتون مدل قسمت قبل به شکل زیر است:

```python
import numpy as np

def SGD_LogisticRegression_kClass(X, t, k, lr=1e-4, max_iter=1e5):
    """
    Attributes
    ----------
    X: 2D-array or matrix
        independent variables
    t: array
        target variables {0,1,....,k-1}
    k: int
        number of classes
    lr: float
        learning rate
    max_iter: int
        maximum number of iteration

    Return
    ------
    w: array
        estimated coefficients
    """

    N, d = X.shape
    T = zeros((N,k))
    for i in range(N):
        T[i][t[i]] = 1
    w = np.zeros((k,d))
    for i in range(max_iter):
        l = i mod N
        for j in range(d):
            w -= lr * (y(w,j,x[l],k) - T[l][j]) * x[l]

    return w


def y(w,ind,x,k):
    return np.exp(w[ind] @ x.T)/ np.sum(np.exp(w @ x.T))
```