Tools and Techniques for ML Homework 3

Zian Jiang (zj444)

April 28, 2021

1 Derivation of importance-weighted reward imputation

1.1

The training dataset is a fixed dataset that is limited to samples from π_0. We lack knowledge of the reward $\delta(X_i, y)$ for many $y \in \mathcal{Y}$ that π would have chosen differently from π_0), but also that the actions preferred by π_0 are over-represented). Thus there is a covariate shift between π_0 and π.

1.2

Due to the covariate shift, we need to remove the distribution mismatch between π_0 and π by adding an importance weight to each term in the squared loss sum.

$$J_{IW}(r) = \frac{1}{n} \sum_{i=1}^{n} \frac{\pi(A_i|X_i)}{\pi_0(A_i|X_i)} (r(X_i, A_i) - R_i(A_i))^2$$

As we can see, $J_{IW}(r)$ is just $J(r)$ with each term re-weighted, thus by the change of measure theorem it follows naturally that $E(J_{IW}(r)) = E(J(r))$.

2 Optimizing 0/1 loss for binary classification

2.1

We can easily write out a table with the 4 cases and their respective losses and see that

$$El(A, Y) = (1-p)\pi + p(1-\pi) = p + \pi - 2p\pi.$$

Take the partial derivative and we can derive that the optimal π is 0.5.

2.2

$$E_{Y\sim Ber(p)} l(\pi, y) = pl(\pi, y = 1) + (1-p)l(\pi, y = 0) = (1-p)\pi + p(1-\pi).$$

Thus, $l(\pi, y = 1) = 1 - \pi$ and $l(\pi, y = 0) = \pi$. According to the hint, we get

$$l(a, y) = a^{1-y}(1-a)^y.$$

2.3

First, note that we can write $P(Y|X = x; w)$ compactly as

$$P(Y|X = x; w) = (\phi(w^T x))^y (1 - \phi(w^T x))^{1-y}.$$

$$E_{X,Y \sim P, a \sim Ber(\phi(w^T x))} \mathbf{1}[a \neq y] = \sum_{a=0}^{1} \sum_{y=0}^{1} P(A = a|X = x; w) P(Y = y|X = x) P(X = x) \mathbf{1}[a \neq y],$$

and canceling further out, we get

$$E_{X,Y \sim P, a \sim Ber(\phi(w^T x))} \mathbf{1}[a \neq y] = (1-\phi(w^T x)) P(Y = 1|X = x) P(X) + \phi(w^T x) P(Y = 0|X = x) P(X).$$

2.4

$$J(w) = \sum_{i=1}^{n} (1 - Y_i) \phi(w^T X_i) + Y_i (1 - \phi(w^T X_i)).$$

This is almost equivalent to the negative log likelihood loss; both functions are optimized when for each example we make a correct prediction. However, consider the case of a large dataset and large feature space, where we have to use stochastic gradient descent (SGD) to optimize. SGD would only work on convex functions and the standard logistic function ϕ is not convex. However, $\log \phi$ is convex. Thus in this case only logistic regression with negative log likelihood loss would perform as expected.