Practice mid-term exam, CS499 Deep Learning Spring 2020

Name: ____________________________ StudentID: ____________________________ 5 Mar

1 Deriving the logistic loss from the binomial likelihood

Logistic regression is a machine learning problem where the output/label $y \in \{0, 1\}$ is binary-valued, and the input/features $x \in \mathbb{R}^n$ is a real vector. We will derive a loss function to use in this case (same as we have seen in class).

We assume $y \sim \text{Bernoulli}(p)$ where $p \in [0, 1]$ is a probability of success / positive label. The probability mass function is

$$\Pr(y, p) = pI[y = 1] + (1 - p)I[y = 0] = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{if } y = 0, \end{cases}$$

where I is the indicator function (1 if argument is true, 0 otherwise).

Derive an expression in terms of y and p for the log-likelihood of the mean parameter p given a single label y:

$\log \Pr(y, p) =$ ____

We learn a **real-valued prediction function,** $f(x) \in \mathbb{R}$ such that $p = \sigma[f(x)]$, where $\sigma : \mathbb{R} \to (0, 1)$ is the logistic link function, $\sigma(z) = 1/(1 + \exp(-z))$.

Now assume for each observation $i \in \{1, \dots, m\}$ we have an input/feature vector $x^{(i)} \in \mathbb{R}^n$ and an output label $y^{(i)} \in \{0, 1\}$ the best function f is the minimizer of the negative log-likelihood,

$$\begin{aligned} \mathcal{L}(f) &= -\sum_{i=1}^{m} \log \Pr(y^{(i)}, \sigma[f(x^{(i)})]) \\ &= \sum_{i=1}^{m} \underline{\hspace{10cm}} \end{aligned}$$

In the blank above, write an expression for the negative log-likelihood in terms of indicator functions and $y^{(i)}, x^{(i)}, f$. To simplify the expression above, we use an alternate definition of the labels,

$$\tilde{y}^{(i)} = \begin{cases} 1 & \text{if } y^{(i)} = 1 \\ -1 & \text{if } y^{(i)} = 0 \end{cases}$$

Write a simplified expression (without indicator functions) for the negative log-likelihood in terms of $\tilde{y}^{(i)}, x^{(i)}, f$:

$$\begin{aligned} \mathcal{L}(f) &= \sum_{i=1}^{m} \log[\underline{\hspace{10cm}}] \\ &= \sum_{i=1}^{m} \ell[f(x^{(i)}), \tilde{y}^{(i)}] \end{aligned}$$

Write the logistic loss function $\ell : \mathbb{R} \times \{-1, 1\} \to \mathbb{R}_+$ for generic predictions $f(x)$ and labels $\tilde{y}$:

$\ell[f(x), \tilde{y}] =$ ____

(total 20 points)

2 Algorithm for un-regularized logistic regression

Logistic regression is a machine learning problem where the output/label $\tilde{y}^{(i)} \in \{-1, 1\}$ is binary-valued, and each input/feature vector $x^{(i)} \in \mathbb{R}^n$ is a real vector as usual. The prediction function $f(x) = w^T x$ depends on a weight vector $w \in \mathbb{R}^n$. Write the total logistic loss function to minimize in terms of $w, x^{(i)}, y^{(i)}$:

$$\mathcal{L}(f) \quad = \quad \mathcal{L}(w) = \sum_{i=1}^{m} \log[\underline{\hspace{10cm}}]$$

What are the types/dimensions of the inputs and the outputs of the gradient function?

$\nabla_w \mathcal{L} : \underline{\hspace{3cm}} \to \underline{\hspace{3cm}}$

Derive an expression in terms of $w, x^{(i)}, y^{(i)}$ for the gradient of the total logistic loss:

$\nabla_w \mathcal{L}(w) = \underline{\hspace{6cm}}$

In L2-regularized logistic regression we want to minimize the penalized cost function,

$$C_\lambda(w, \beta) = \mathcal{L}(w, \beta) + R_\lambda(w)$$

where the regularization term involves the sum of squares of the components of the weight vector, $R_\lambda(w) = \lambda ||w||_2^2 = \lambda \sum_{j=1}^{p} w_j^2$.

Write $\nabla_w R_\lambda(w) = \underline{\hspace{6cm}}$ in terms of λ and w.

Derive $\nabla_w C_\lambda(w) = \underline{\hspace{6cm}}$

The goal of gradient descent is to compute $\arg\min_{w \in \mathbb{R}^p} C_\lambda(w)$. Fill in the blank in the gradient step below, in terms of the functions defined above. Use a constant step size α.

```
1: $w \leftarrow$ ______________________________
```

(total 30 points)

3 K-fold cross-validation

The image below represents a data set with $n = 70$ observations, one for each individual image of a digit. Say we want to determine which of several different machine learning models (e.g. linear model with early stopping, nearest neighbors, etc) is most accurate in these data. To do that we perform 3-fold cross-validation. Fold ID numbers $\in \{1, 2, 3\}$ have been assigned to all observations/images in the corresponding row/letter.

A, fold=1

B, fold=2

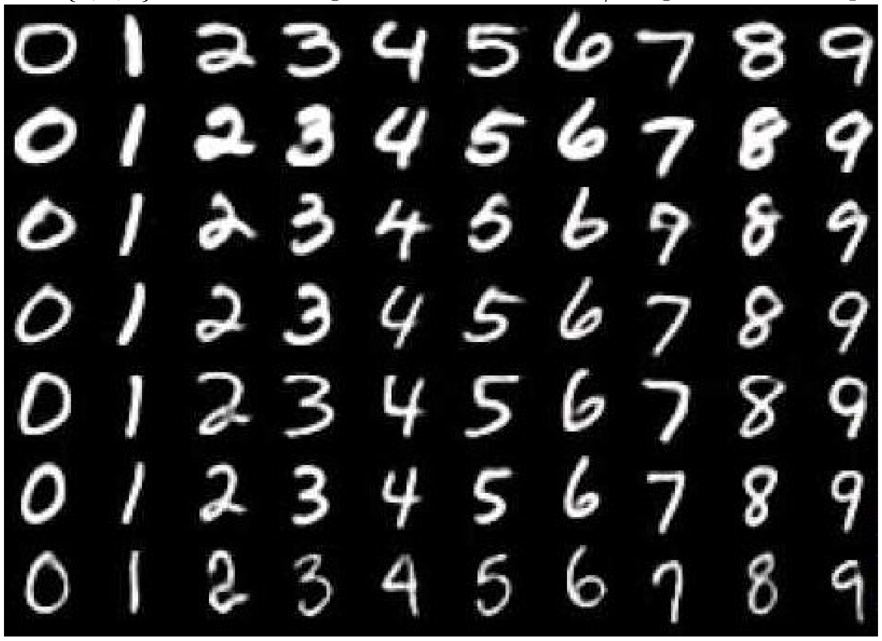
C, fold=3

D, fold=1

E, fold=2

F, fold=1

G, fold=3



1. For fold/split 1 which observations/letters
   are the train set which are passed to the learning algorithm/function? ____________________

   For fold/split 1 which observations/letters are used for test set? (learning algorithm can not access these data, but the learned model is used to predict for them) ____________________

2. For fold/split 2. Train set = ____________________, Test set = ____________________.

3. For fold/split 3. Train set = ____________________, Test set = ____________________.

4. Now assume that we are in the context of fold/split 1. Your learning algorithm has access to all of the data in the train set. Your learning algorithm uses 2-fold cross-validation internally to select the best regularization parameter η (e.g. steps, neighbors, penalty, etc). To do these subtrain/validation splits, randomly assign each observation/letter in the train set to a new/internal fold ID.

   Fold 1 = ____________________, Fold 2 = ____________________

5. Now assume that your function has computed MeanValidationLoss(η), the mean validation loss over both validation folds, for r regularization parameters $\eta_1, \dots, \eta_r$. What is the best model/regularization parameter η^* that you should select to make the final predictions on the test set for fold 1?

$$\eta^* = \rule{6cm}{0.4pt}$$

(total 20 points)

4 Neural network learning algorithm

In a single-layer neural network the prediction function is

$$f(x^{(i)}) = b^{(i)} = w^T z^{(i)} = w^T \sigma(a^{(i)}) = w^T S(V^T x^{(i)})$$

The feature matrix is $X \in \mathbb{R}^{m\times n}$, the feature vector for one observation is $x^{(i)} \in \mathbb{R}^n$. Labels are $y \in \mathbb{R}^m$ for regression. One label is $y^{(i)} \in \mathbb{R}$. The hidden layer vector is $z^{(i)} = S(a^{(i)}) = S(V^T x^{(i)}) \in \mathbb{R}^u$. One hidden unit is $z_k^{(i)} = \sigma(a_k^{(i)}) = 1/(1+\exp(-a_k^{(i)})) \in \mathbb{R}$. Indices are $i \in \{1,\dots,m\}$ for observations/examples, $j \in \{1,\dots,n\}$ for features, and $k \in \{1,\dots,u\}$ for hidden units. The weight matrix for the first layer is $V \in \mathbb{R}^{n\times u}$, and the weight vector for predicting hidden unit k is $v_k \in \mathbb{R}^n$. The weight vector for predicting the output is $w \in \mathbb{R}^u$. The overall loss function that we want to minimize is

$$\mathcal{L}(w,V) = \frac{1}{m}\sum_{i=1}^m \frac{1}{2}[f(x^{(i)}) - y^{(i)}]^2 = \frac{1}{m}\sum_{i=1}^m \frac{1}{2}[w^T S(V^T x^{(i)}) - y^{(i)}]^2 = \frac{1}{2m}||S(XV)w - y||_2^2$$

The learning algorithm is as before with linear models: start at $w, V = 0$ (or some random values close) and then take steps in the opposite direction of the gradient. We therefore need to compute the gradients of $\mathcal{L}$ with respect to the parameters w, V. We consider the gradient with respect to a single observation i:

$$\nabla_w \frac{1}{2}[f(x^{(i)}) - y^{(i)}]^2 = \underbrace{\frac{\partial}{\partial b^{(i)}}\frac{1}{2}[b^{(i)} - y^{(i)}]^2}_{\delta^{w(i)}} \nabla_w b^{(i)}$$

And we consider the gradient of the weights $v_k \in \mathbb{R}^n$ used to predict hidden unit k:

$$\nabla_{v_k} \frac{1}{2}[f(x^{(i)}) - y^{(i)}]^2 \quad = \quad \underbrace{\frac{\partial}{\partial a_{ik}}\frac{1}{2}[w^T S(a^{(i)}) - y^{(i)}]^2}_{\delta^v_{ik}} \nabla_{v_k} a_{ik}$$

The algorithm is to compute, in order: (3 points each)

quantity	to compute	in terms of
hidden before sigmoid	$a^{(i)} =$ ______________________________ $\in \mathbb{R}^u$	$V, x^{(i)}$
hidden after sigmoid	$z^{(i)} =$ ______________________________ $\in \mathbb{R}^u$	$S, a^{(i)}$
predictions	$b^{(i)} =$ ______________________________ $\in \mathbb{R}$	$w, z^{(i)}$
second level errors	$\delta^{(i)w} =$ ______________________________ $\in \mathbb{R}$	$b^{(i)}, y^{(i)}$
first level errors	$\delta^{(i)v} =$ ______________________________ $\in \mathbb{R}^u$	$\delta^{(i)w}, w, S, a^{(i)}$
second level gradient	$\nabla_w \frac{1}{2}[f(x^{(i)}) - y^{(i)}]^2 =$ ______________________________ $\in \mathbb{R}^u$	$\delta^{(i)w}, z^{(i)}$
first level gradient	$\nabla_V \frac{1}{2}[f(x^{(i)}) - y^{(i)}]^2 =$ ______________________________ $\in \mathbb{R}^{n\times u}$	$\delta^{(i)v}, x^{(i)}$

(total 30 points)