CS 541 Deep Learning | **Assignment 1** | 2023-01-27

Vaibhav Nandkumar Kadam
vkadam@wpi.edu

Aadesh Varude
avarude@wpi.edu

Q1 . Solution.
Following is method for Double Cross Validation on Dataset D with folds k and hyperparameter set H.

Please find doDoubleCrossValidation.py for the source code.

```
def doCrossValidation (D, k, h):

    allIdxs = np.arange(len(D))
    # Randomly split dataset into k folds
    idxs = np.random.permutation(allIdxs)
    idxs = idxs.reshape(k, -1)
    accuracies = []
    h_best=INT_MAX
    acc=0
    for fold in range(k):
        # Get all indexes for this fold
        testIdxs = idxs[fold,:]
        # Get all the other indexes
        trainIdxs = np.array(set(allIdxs) - set(testIdxs)).flatten()
        # Train the model on the training data
        model = trainModel(D[trainIdxs], h)
        accuracy=testModel(model, D[testIdxs])
        if accuracy>acc:
            acc=accuracy
            h_best=h

    return h_best

# H is the list of list of hyperparameters


def doDoubleCrossValidation(D, k, H):

    allIdxs = np.arange(len(D))
    # Randomly split dataset into k folds
    idxs = np.random.permutation(allIdxs)
    idxs = idxs.reshape(k, -1)

    accuracies = []

    #Outer loop
    for outer_fold in range(k):
        # Get all indexes for this fold
        testIdxs_outer = idxs[outer_fold,:]
        # Get all the other indexes
        trainIdxs_outer = idxs[list(set(allIdxs) - set(testIdxs)),:].flatten()
        # Train the model on the training data


        h_best = doCrossValidation(D[trainIdxs_outer], k, H[outer_fold])
        accuracies.append(testModel(model, D[testIdxs_outer]))

    return np.mean(accuracies)
```

Q2 a. Solution.

We have $f(x, y) = x^4 + xy + x^2$
we take,

$$\frac{\partial f}{\partial x} = 4x^3 + y + 2x \tag{1}$$

$$\frac{\partial f}{\partial y} = x \tag{2}$$

We calculate Hessian matrix H_f

$$H_f = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x \partial x} & \dfrac{\partial^2 f}{\partial x \partial y} \\ \dfrac{\partial^2 f}{\partial y \partial x} & \dfrac{\partial^2 f}{\partial y \partial y} \end{bmatrix} \tag{3}$$

$$H_f = \begin{bmatrix} 12x^2 + 2 & 1 \\ 1 & 0 \end{bmatrix} \tag{4}$$

So,

$$\det(H_f) = -1 \tag{5}$$

We know that the determinant of the matrix is the multiplication of all eigen values.
As the determinant is negative at least one eigen value should be -ve. hence we can say that $\lambda_j \ngeq 0$ which proves that the matrix is not PD or PSD as $\det(H_f)$ is -1.
Thus as the $\det(H_f)$ is -1 the given function is not convex for $\in \mathbb{R}$

Q2 b. Solution.

We have f_{mse} for two layer neural network ,

$$f_{mse} = \frac{1}{2}(X^T w - y)^T (X^T w - y) \tag{6}$$

y is the n-dimensional vector of ground-truth labels and X is the design matrix and weight vector w.

For simplifying we ignore the $\frac{1}{2n}$ for now but will use it later.

$$f_{mse} = (X^T w - y)^T (X^T w - y) \tag{7}$$

From 3.d in HW 1 we know

$$\nabla_x [(Ax + b)^T (Ax + b)] = 2A^T (Ax + b) \tag{8}$$

Similarly we get,

$$\nabla_w [(x^T w - y)^T (x^T w - y)] = 2(X^T)^T X^T w \tag{9}$$

We get,

$$\nabla_w = 2XX^T w \tag{10}$$

Now taking the Hessian H of the f_{mse} i.e ∇^2_w

Thus we get

$$H = 2XX^T \tag{11}$$

Considering $\frac{1}{2n}$ from eq. 5 we get,
we get

$$H = \frac{1}{n} XX^T \tag{12}$$

For any vector $v \in \mathbb{R}$,

$$v^T H v = v^T (H) v \tag{13}$$

$$v^T H v = v^T (XX^T) v \tag{14}$$

$$v^T H v = (Xv)^T (Xv) \tag{15}$$

$$v^T H v = \|Xv\|_2^2 \geq 0 \tag{16}$$

Therefore we have,

$$v^T H v \geq 0 \tag{17}$$

Thus the given Hessian matrix H is Positive semi definite and $v^T H v \geq= 0$ for any real vector v.

Q3 . Solution.

```
# The best hyperparameters after training are
# Learning rate: 0.001
# Epochs: 300
# Batchsize: 16
# Alpha:0.5
# lr_f,bs_f,epochs_f,alpha_f=0.001, 16 ,300, 0.5
```

The final MSE loss for the test data set is Dte is 83.99095638911503.

Q4 . Solution.

We have

$$\sigma(x) = \frac{1}{1+e^{-x}} \forall x \tag{18}$$

Solution a.

$$\sigma(-x) = \frac{1}{1+e^{x}} \tag{19}$$

So,

$$\begin{aligned} 1-\sigma(-x) &= 1-\frac{1}{1+e^{x}} \\ &= \frac{1+e^{x}-1}{1+e^{x}} \\ &= \frac{e^{x}}{1+e^{x}} \\ &= \frac{1}{1+e^{-x}} \end{aligned} \tag{20}$$

From Eq. 18 and 20 we get

$$\sigma(-x) = 1-\sigma(x) \tag{21}$$

Hence Proved.

Solution b.

For calculating $\sigma\prime(x)$ we take

$$\begin{aligned} \frac{\partial\sigma(x)}{dx} &= \frac{\partial}{dx}(\frac{1}{1+e^{-x}}) \\ &= \frac{\partial}{\partial x}(1+e^{x})^{-1} \\ &= -(1+e^{-x})^{-2}\cdot(-e^{-x}) \\ &= \frac{e^{-x}}{(1+e^{-x})^{2}} \\ &= \frac{1}{1+e^{-x}}\cdot\frac{e^{-x}}{1+e^{-x}} \end{aligned} \tag{22}$$

We get ,

$$\frac{\partial\sigma(x)}{dx} = \sigma(x)\cdot\sigma(-x) \tag{23}$$

From equation 21 we get,

$$\sigma\prime(x) = \sigma(x)\cdot(1-\sigma(x)) \tag{24}$$

Hence Proved.
