CENG783 Deep Learning HW1 - Written Solutions
Ege Özbek 2237709
ege.ozbek@metu.edu.tr

Q1.2.1

To help us with differentiation lets call
$z = wx + b$
$f(x; w, b) = p$
$s(y, p) = y - p$
$L(s) = \frac{1}{N} \sum_{i=1}^{N} s^2$

Partial derivative of Loss w.r.t w can be calculated using chain rule

$\frac{\partial}{\partial w} L(w, b) = \frac{\partial L}{\partial s} * \frac{\partial s}{\partial w}$
$\frac{\partial L}{\partial s}$ is $\frac{\partial L}{\partial s} = \frac{2}{N} \sum_{i=1}^{N} s$

S can be decompositioned as
$\frac{\partial s}{\partial w} = \frac{\partial y}{\partial w} - \frac{\partial p}{\partial w}$ where $\frac{\partial y}{\partial w} = 0$

Partial derivative simplifies into : $\frac{\partial s}{\partial w} = -\frac{\partial p}{\partial w}$

Linear operation is p(z)
$p(z) = 0 \quad if \quad z \leq -1$
$\quad z/2 + 0.5 \quad if \quad -1 < z < -1$
$\quad 1 \quad if \quad z \geq 1$

Derivative of p is $\frac{\partial p}{\partial z} = 0 \quad if \quad z \leq -1$
$\quad 0.5 \quad if \quad -1 < z < -1$
$\quad 1 \quad if \quad z \geq 1$

Z = WX
$\frac{\partial z}{\partial w} = 0 \quad if \quad z \leq -1$
$\quad X^T \quad if \quad -1 < z < -1$
$\quad 0 \quad if \quad z \geq 1$

Partial derivative of activation function w.r.t to w is
$\frac{\partial p}{\partial w} = 0 \quad if \quad z \leq -1$
$\quad \frac{x^T}{2} \quad if \quad -1 < z < -1$
$\quad 0 \quad if \quad z \geq 1$

Combing the chain rule derivatives , total equation simplifies to
$\frac{\partial L}{\partial w} = 0 \quad if \quad wx + b \leq -1$
$\quad \frac{1}{N} \sum_{i=1}^{N} (f(x; w, b) - y) \, X^T \quad if \quad -1 < wx + b < -1$
$\quad 0 \quad if \quad wx + b \geq 1$

Q1.2.2

Partial derivative of Loss w.r.t w can be calculated in a similar fashion,
$\frac{\partial}{\partial b} L(w, b) = \frac{\partial L}{\partial s} * \frac{\partial s}{\partial b}$
$\frac{\partial s}{\partial b} = \frac{\partial y}{\partial b} - \frac{\partial p}{\partial b}$ where $\frac{\partial y}{\partial b} = 0$

Derivative of Linear operation can be described as,
$\frac{\partial z}{\partial b} = 0 \quad if \quad z \leq -1$
$\quad 1 \quad if \quad -1 < z < -1$

$0 \quad if \quad z \geq 1$

Partial derivative of activation function w.r.t to w is,
$\frac{\partial p}{\partial b} = 0 \quad if \quad z \leq -1$
$\frac{1}{2} \quad if \quad -1 < z < -1$
$0 \quad if \quad z \geq 1$

Comibing the chain rule derivatives , total equation simplifies to
$\frac{\partial L}{\partial b} = 0 \quad if \quad wx + b \leq -1$
$\frac{1}{N} \sum_{i=1}^{N} (f(x; w, b) - y) \quad if \quad -1 < wx + b < -1$
$0 \quad if \quad wx + b \geq 1$

Q.1.4

To prove that L2Loss is convex we need to show that it is composed of convexity preserving operations

$L(x; w, b) = \frac{1}{N} \sum_{i=1}^{N} (y - f(x; w, b))^2 \, between \quad -1 \leq wx + b \leq 1$
and = 0 otherwise

$\frac{1}{N}$can be simplified out since it is an affine operation,
$\sum_{i=1}^{N}$is also preserves convexity since it also counts as an affine operation,
Square operation also preserves convexity thus we only need to focus on the operations inside$(y - f(x; w, b))^2$
Y(the label) is just a constant, all we need to show is that $f(x; w, b)$ conserves convexity in its domain (-1,1)
$p(z) = z/2 + 0.5 \quad if \quad -1 < z < -1$
substituting Z for WX+b
$p(WX + b) = \frac{WX+b}{2} + 0.5 \quad if \quad -1 < WX + b < -1$
Matrix multiplication is basically a weighted sum and therefore ensures convexity, this ensures us that
L2Loss is convex in terms of W and b.

Q.2.1

Basically we divide dataset into k different splits, at each round we leave one split out for testing and train the model on the rest of the dataset,

At each round we keep track of the models performance by testing it on the split that is left out for testing.

After k repeats , we would keep the model with best trained parameters (who has the best C parameter).

Pseudo-code:

```
train_split , test_split = divide_dataset(dataset)
for each subset of S:
   model = train(train_split)
   test_score = test(model,test_split)
   if test_score > previousBestScore:
      previousBestScore = test_score
      bestModel = model(with the best C)
```

Q.2.2

To estimate a model's future performance on unseen data, we require a test dataset, on which no kind of training or fine-tuning will take place.

This test dataset gives us a clean benchmark on unseen data since model has never been subjected to it.

In practice we can do a 60-20-20 split where %60 of dataset is allocated for training purposes, %20 is left out for fine-tuning hyperparameters(cross-validation set) and %20 remains for testing against unseen examples. This approach would work since we know that unseen data comes from the same population(distribution) as the data that is used for cross-validation, we can surely expect any fine tuning done by cross-validation dataset to have a positive effect on score achieved on unseen data.

Pseudo-code:

```
training_set,validation_set,test_set = split_dataset(dataset)
```

```
model = train(training_set)
hyper_parameters = fine_tune(model,validation_set)
test_score = test(model , test_set)
```