AI6102: Machine Learning Methodologies & Applications

Ong Jia Hui (G1903467L)
JONG119@e.ntu.edu.sg

Assignment 1

1 Question One

Consider a multi-class classification problem of C classes. Based on the parametric forms of the conditional probabilities of each class introduced on the 54th Page ("Extension to Multiple Classes") of the lecture notes of L4, derive the learning procedure of logistic regression for multi-class classification problems. *Hint: define a loss function by borrowing an idea from binary classification, and derive the gradient descent rules to update* $\{\mathbf{w}^{(c)}\}$.

1.1 Answers

1.1.1 Multi-nominal Logistic Regression Approach

Suppose there are C classes, {0, 1, ..., C-1}. Each class except class 0 is associated with a specific $w^{(c)}$, $c = 1, ..., C-1$:

$$\textit{For } c > 0\textit{:} \quad P(y=c|x) = \frac{\exp\left(-w^{(c)^T}x\right)}{1+\sum_{c=1}^{C-1}\exp\left(-w^{(c)^T}x\right)} = \hat{y}_c \tag{1}$$

$$\textit{For } c = 0\textit{:} \quad P(y=0|x) = \frac{1}{1+\sum_{c=1}^{C-1}\exp\left(-w^{(c)^T}x\right)} = \hat{y}_0 \tag{2}$$

$$\textit{where} \quad \sum_{c=0}^{C-1} P(y=c|x) = 1$$

For simplicity of annotation, we denote Equation 1 as $\hat{y}_c$ and Equation 2 as $\hat{y}_0$ such that the binary cross entropy (BCE) loss can be written as:

$$L(w) = \sum_{c=1}^{C-1} y_c \ln \hat{y}_c + y_0 \ln(1-\hat{y}_c) \tag{3}$$

Given N training examples, where input features are $x_i \in \mathbb{R}^M$. The total loss function to minimize is:

$$J(w) = -\sum_{i=1}^{N}\left[\sum_{c=1}^{C-1} y_{ci}\ln\hat{y}_{ci} + y_{0i}\ln\hat{y}_{0i}\right] + \frac{\lambda}{2}\|\mathbf{w}\|_2^2 \tag{4}$$

Based on Gradient Descent update rule (where α is the learning rate), we need to compute the 1st order derivative of the loss function w.r.t the weights:

$$\begin{aligned} w_{t+1} &= w_t - \alpha\frac{\partial J(w)}{\partial w} \\ w_{t+1} &= w_t - \alpha\left(-\sum_{i=1}^{N}\frac{\partial L(w)}{\partial w} + \frac{\partial \frac{\lambda}{2}\|\mathbf{w}\|_2^2}{\partial w}\right) \end{aligned} \tag{5}$$

To derive this gradient $\frac{\partial L(w)}{\partial w}$, we can use the Chain Rule, where we let $z^{(i)} = -{w^{(i)}}^T x_i$:

$$\frac{\partial L(w)}{\partial w} = \frac{\partial L(w)}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial z} \times \frac{\partial z}{\partial w}$$

We first find the derivative for $L(w)$ w.r.t to $z^{(i)}$ without the regulation term:

$$\begin{aligned}
\frac{\partial L(w)}{\partial z^{(i)}} &= \frac{\partial \left[\sum_{c=1}^{C-1} y_c \ln \hat{y}_c + y_0 \ln \hat{y}_0\right]}{\partial z^{(i)}} \\
&= \frac{\partial}{\partial z^{(i)}} \left[\sum_{c=1}^{C-1} y_c \ln \hat{y}_c\right] + \frac{\partial \left(y_0 \ln \hat{y}_0\right)}{\partial z^{(i)}} \\
&= \sum_{c=1}^{C-1} y_c \frac{\partial \ln \hat{y}_c}{\partial z^{(i)}} + y_0 \frac{\partial \ln \hat{y}_0}{\partial z^{(i)}} \\
&= \sum_{c=1}^{C-1} \frac{y_c}{\hat{y}_c} \frac{\partial \hat{y}_c}{\partial z^{(i)}} + \frac{y_0}{\hat{y}_0} \frac{\partial \hat{y}_0}{\partial z^{(i)}}
\end{aligned} \tag{6}$$

From Equation 6, we need to find $\frac{\partial \hat{y}_c}{\partial z^{(i)}}$ and $\frac{\partial \hat{y}_0}{\partial z^{(i)}}$.
We start with finding $\frac{\partial \hat{y}_c}{\partial z^{(i)}}$:

$$\frac{\partial \hat{y}_c}{\partial z^{(i)}} = \frac{\partial}{\partial z^{(i)}} \left(\frac{\exp\left(-{w^{(c)}}^T x\right)}{1 + \sum_{c=1}^{C-1} \exp\left(-{w^{(c)}}^T x\right)} \right)$$

For derivative of $\hat{y}_c$, we need to use the Quotient Rule.

The Quotient Rule states that $f'(x) = \frac{g'(x)h(x) - h'(x)g(x)}{(h(x))^2}$.
Before applying Quotient Rule, we find the partial derivatives for $\frac{g(x)}{h(x)}$ separately:

$$\begin{aligned}
\mathit{Let}: \quad g(x) &= \exp\left(z^{(c)}\right) = \exp\left(w^{-(c)^T} x_i\right) \\
\mathit{Let}: \quad h(x) &= \sum_{c=1}^{C-1} \exp\left(z^{(c)}\right) = 1 + \sum_{c=1}^{C-1} \exp\left(-{w^{(c)}}^T x_i\right)
\end{aligned}$$

$$g'(x) = \frac{\partial g(x)}{\partial z^{(i)}} = \frac{\partial \exp\left(-{w^{(c)}}^T x_i\right)}{\partial (-{w^{(i)}}^T x_i)} = \begin{cases} \exp\left(-{w^{(i)}}^T x_i\right) & i = c \\ 0 & i \neq c \end{cases}$$

$$h'(x) = \frac{\partial h(x)}{\partial z^{(i)}} = \frac{\partial \left[1 + \sum_{c=1}^{C-1} \exp\left(-{w^{(c)}}^T x_i\right)\right]}{\partial (-{w^{(i)}}^T x_i)} = \exp\left(-{w^{(i)}}^T x_i\right)$$

Hence, for the case when $i = c$:

$$\begin{aligned}
\frac{\partial \hat{y}_c}{\partial z^{(i)}} &= \frac{\exp\left(-w^{(i)^T} x_i\right)\left[1 + \sum_{c=1}^{C-1} \exp\left(-w^{(c)^T} x_i\right)\right] - \exp\left(-w^{(i)^T} x_i\right)\exp\left(-w^{(c)^T} x_i\right)}{\left(1 + \sum_{c=1}^{C-1} \exp(-w^{(c)^T} x_i)\right)^2} \\
&= \frac{\exp\left(-w^{(i)^T} x_i\right)\left[1 + \sum_{c=1}^{C-1} \exp\left(-w^{(c)^T} x_i\right) - \exp\left(w^{(c)^T} x_i\right)\right]}{\left(1 + \sum_{c=1}^{C-1} \exp(-w^{(c)^T} x_i)\right)^2} \\
&= \frac{\exp\left(-w^{(i)^T} x_i\right)}{1 + \sum_{c=1}^{C-1} \exp(-w^{(c)^T} x_i)} \; \frac{1 + \sum_{c=1}^{C-1} \exp\left(-w^{(c)^T} x_i\right) - \exp\left(-w^{(c)^T} x_i\right)}{1 + \sum_{c=1}^{C-1} \exp(-w^{(c)^T} x_i)} \\
&= \frac{\exp\left(-w^{(i)^T} x_i\right)}{1 + \sum_{c=1}^{C-1} \exp(-w^{(c)^T} x_i)} \; \left(\frac{1 + \sum_{c=1}^{C-1} \exp\left(-w^{(c)^T} x_i\right)}{1 + \sum_{c=1}^{C-1} \exp(-w^{(c)^T} x_i)} - \frac{\exp\left(-w^{(c)^T} x_i\right)}{1 + \sum_{c=1}^{C-1} \exp(-w^{(c)^T} x_i)}\right) \\
&= \hat{y}_i \left(1 - \hat{y}_c\right) \\
&= \hat{y}_i \left(1 - \hat{y}_i\right) \qquad (7)
\end{aligned}$$

Then, for the case when $i \neq c$:

$$\begin{aligned}
\frac{\partial \hat{y}_c}{\partial z^{(i)}} &= \frac{(0)\left(1 + \sum_{c=1}^{C-1} \exp\left(-w^{(c)^T} x_i\right)\right) - \exp\left(-w^{(i)^T} x_i\right)\exp\left(-w^{(c)^T} x_i\right)}{\left(1 + \sum_{c=1}^{C-1} \exp(-w^{(c)^T} x_i)\right)^2} \\
&= -\frac{\exp\left(-w^{(i)^T} x_i\right)\exp\left(-w^{(c)^T} x_i\right)}{\left(1 + \sum_{c=1}^{C-1} \exp(-w^{(c)^T} x_i)\right)^2} \\
&= -\frac{\exp\left(-w^{(i)^T} x_i\right)}{1 + \sum_{c=1}^{C-1} \exp(-w^{(c)^T} x_i)} \; \frac{\exp\left(-w^{(c)^T} x_i\right)}{1 + \sum_{c=1}^{C-1} \exp(-w^{(c)^T} x_i)} \\
&= -\hat{y}_i \left(\hat{y}_c\right) \qquad (8)
\end{aligned}$$

Therefore with Equations 7 and 8, the derivatives of $\frac{\partial \hat{y}_c}{\partial z^{(i)}}$ are:

$$\frac{\partial \hat{y}_c}{\partial z^{(i)}} = \begin{cases} \hat{y}_i \left(1 - \hat{y}_i\right) & i = c \\ -\hat{y}_i \left(\hat{y}_c\right) & i \neq c \end{cases}$$

Now we find the derivative for $\frac{\partial \hat{y}_0}{\partial z^{(i)}}$:

$$\begin{aligned}
\frac{\partial \hat{y}_0}{\partial z^{(i)}} &= \frac{\partial}{\partial z^{(i)}} \left(\frac{1}{1 + \sum_{c=1}^{C-1} \exp(-w^{(c)^T} x)}\right) \\
&= (-1)\frac{\exp\left(-w^{(i)^T} x\right)}{\left(1 + \sum_{c=1}^{C-1} \exp(-w^{(c)^T} x)\right)^2} \\
&= -\frac{1}{1 + \sum_{c=1}^{C-1} \exp(-w^{(c)^T} x)} \frac{\exp\left(-w^{(i)^T} x\right)}{1 + \sum_{c=1}^{C-1} \exp(-w^{(c)^T} x)} \\
&= -\hat{y}_0 \left(\hat{y}_i\right) \qquad (9)
\end{aligned}$$

Sub the derivatives found in Equations 7, 8 and 9 back into Equation 6:

$$\begin{aligned}
\frac{\partial L(w)}{\partial z^{(i)}} &= \sum_{c=1}^{C-1} \frac{y_c}{\hat{y}_c}\frac{\partial \hat{y}_c}{\partial z^{(i)}} + \frac{y_0}{\hat{y}_0}\frac{\partial \hat{y}_0}{\partial z^{(i)}} \\
&= \left[\sum_{c=1,c\neq i}^{C-1} \frac{y_c}{\hat{y}_c}\frac{\partial \hat{y}_c}{\partial z^{(i)}} + \frac{y_i}{\hat{y}_i}\frac{\partial \hat{y}_i}{\partial z^{(i)}}\right] + \frac{y_0}{\hat{y}_0}\frac{\partial \hat{y}_0}{\partial z^{(i)}} \\
&= \sum_{c=1,c\neq i}^{C-1} \frac{y_c}{\hat{y}_c}\left(-\hat{y}_i\hat{y}_c\right) + \frac{y_i}{\hat{y}_i}\left[\hat{y}_i\left(1-\hat{y}_i\right)\right] + \frac{y_0}{\hat{y}_0}\left(-\hat{y}_0\hat{y}_i\right) \\
&= \sum_{c=1,c\neq i}^{C-1} \left(-y_c\hat{y}_i\right) + y_i\left(1-\hat{y}_i\right) + \left(-y_0\hat{y}_i\right) \\
&= \sum_{c=1,c\neq i}^{C-1} \left(-y_c\hat{y}_i\right) + y_i - y_i\hat{y}_i - \left(y_0\hat{y}_i\right) \\
&= y_i - \hat{y}_i\left[\sum_{c=1,c\neq i}^{C-1}\left(y_c\right) + y_i + y_0\right] \\
&= y_i - \hat{y}_i\left[\sum_{c=0}^{C-1}\left(y_c\right)\right] \\
&= y_i - \hat{y}_i\left(1\right) \\
&= y_i - \hat{y}_i
\end{aligned} \tag{10}$$

The derivatives of $z^{(i)}$ w.r.t to the weights $w^{(i)}$ is:

$$\begin{aligned}
\frac{\partial z^{(i)}}{\partial w^{(i)}} &= \frac{\partial(-{w^{(i)}}^T x_i)}{\partial w^{(i)}} \\
&= -x_i
\end{aligned}$$

Hence,

$$\begin{aligned}
\frac{\partial L(w)}{\partial w^{(i)}} &= \frac{\partial L(w)}{\partial z^{(i)}} \times \frac{\partial z^{(i)}}{\partial w^{(i)}} \\
&= \left(y_i - \hat{y}_i\right)\left(-x_i\right) \\
&= \left(\hat{y}_i - y_i\right)x_i
\end{aligned} \tag{11}$$

Now we find the derivative for the regularization term, $\frac{\lambda}{2}\|\mathbf{w}\|_2^2$:

$$\frac{\partial \frac{\lambda}{2}\|\mathbf{w}\|_2^2}{\partial w} = \lambda w \tag{12}$$

Sub Equations 11 and 12 into Gradient Descent rule from Equation 5:

$$\begin{aligned}
w_{t+1} &= w_t - \alpha\left(-\sum_{i=1}^{N}\frac{\partial L(w)}{\partial w} + \frac{\partial \frac{\lambda}{2}\|\mathbf{w}\|_2^2}{\partial w}\right) \\
w_{t+1} &= w_t + \alpha\left(\sum_{i=1}^{N}\left(\hat{y}_i - y_i\right)x_i - \lambda w\right)
\end{aligned} \tag{13}$$

2 Question Two

This is a hands-on exercise to use the LinearSVC API of scikit-learn to train a linear SVM on a binary classification dataset.

1. Download the a5a dataset from the LIBSVM Dataset page and load into a sparse data format.
2. Given a set of 5 candidate values of parameter C in LinearSVC, use 3-fold cross-validation to determine C^*.
3. Use LinearSVC to train a model using the whole a5a training set with C^* and make predictions on the a5a test set to find the score accuracy.

2.1 Answers

2.1.1 Configuration

This section lists the libraries (Listing 1) used as well as the variables and constants (Listing 2) defined.

Listing 1: Imported libraries

```
import os
import numpy as np
import sklearn
from matplotlib.pyplot import plt
from sklearn.datasets import load_svmlight_file
from sklearn import svm
from sklearn.model_selection import cross_val_score
```

Listing 2: Constants and variables

```
DATA_PATH = '../data/'
TRAIN_FILE = 'a5a.txt'
TEST_FILE = 'a5a.t.txt'
N_FEATURES = 123     # Number of features
K_FOLD = 3           # 3-Fold CV
C_values = [0.01, 0.1, 1, 10, 100] # Candidate Cs
```

2.1.2 Download and Load Dataset

Listing 3: Get data from file and load into sparse matrix

```
def get_data(filename):
    data = load_svmlight_file(filename, n_features=N_FEATURES) # Fix to 123
    return data[0], data[1]

X_train, y_train = get_data(DATA_PATH + TRAIN_FILE)
X_test, y_test = get_data(DATA_PATH + TEST_FILE)
```

From the download page, the following dataset information can be obtained:

- Num of classes: 2 (-1 and 1)
- Num of data: 6,414 (train) / 26,147 (test)
- Num of features: 123 (train) / 123 (test)

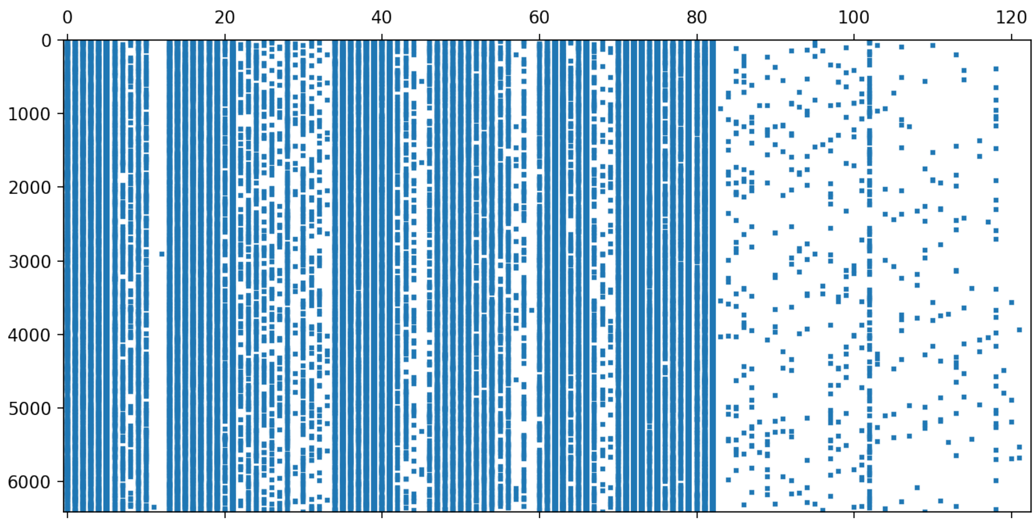

Figure 1: Visualization of X_train's Sparse CSR Matrix using 'matplotlib.pyplot.spy()' method

2.1.3 Determine best C^* using 3-fold CV

Listing 4: Loop through all values of C (Listing 2) while doing 3-fold CV using LinearSVC

```
accuracies = dict.fromkeys(C_values)
for c_ in C_values:
    clf = svm.LinearSVC(C=c_, max_iter=60000) #60,000 to avoid convergence warning
    scores = cross_val_score(clf, X_train, y_train, cv=K_FOLD, scoring='accuracy')
    accuracies[c_] = scores.mean()
    print("C: \%0.2f Accuracy: \%0.5f (+/- \%0.5f)"\%(c_, scores.mean(), scores.std()*2))
```

Table 1: Accuracy of Linear SVCs using various C values with 3-fold CV

C	0.01	0.1	1	10	100
Accuracy [default] (max_iter=1000)	0.83957 (+/- 0.01153)	**0.84222** (+/- 0.01228)	0.83801 (+/- 0.00817)	0.83817 (+/- 0.00939)	0.79732 (+/- 0.03390)
Accuracy (max_iter=60,000)	0.83957 (+/- 0.01153)	**0.84222** (+/- 0.01228)	0.83801 (+/- 0.00817)	0.83645 (+/- 0.00841)	0.83630 (+/- 0.00861)

Table 1 shows the 3-fold cross validation results of various C values using LinearSVC. When using default LinearSVC's settings of max_iter=1000 with C values of 10 and 100, a warning message showing convergence error will appear. It seems that after setting the max_iter to 60,000, the LinearSVC models for C values of 10 and 100 will then manage to converge. Nonetheless, the best value for hyperparameter, C^* is found to be **0.1** regardless of max_iter values.

2.1.4 Train LinearSVC model with C^* and Evaluate on test set

Listing 5: Train LinearSVC using c* = 0.1 and evaluate on test set

```
best_c = max(accuracies, key=accuracies.get) # get best c
svm_clf = svm.LinearSVC(C=best_c)
svm_clf.fit(X_train, y_train)
# predictions = svm_clf.predict(X_test)
accuracy = svm_clf.score(X_test, y_test)
print(f'Accuracy of LinearSVC with C: {best_c} is {accuracy}')
```

Table 2: Evaluation result on testset using LinearSVC trained with c*

C	c*=0.1
Accuracy of linear SVMs	0.84587

The accuracy score of the LinearSVC trained with $C^* = 0.1$ on training set on test set is 0.845871419283283.

3 Question Three (Optional)

Using the kernel trick introduced in L5 to extend the regularized linear regression model to solve nonlinear regression problems. Derive a closed-form solution.

3.1 Answers

Given the regularized linear regression formula in slide 39 of lecture note 3 as follows:

$$\hat{\mathbf{w}} = \arg\min_{w} \quad \frac{1}{2}\sum_{i=1}^{N}(y_i - \mathbf{w}\cdot x_i)^2 + \frac{\lambda}{2}\|\mathbf{w}\|_2^2$$

We know that for any solution $\hat{\mathbf{w}}$, it can be written as a linear combination of features x_i:

$$\hat{w} = \sum_{i=1}^{N}\mathbf{w}\cdot x_i = \sum_{i=1}^{N}\beta_i x_i$$

Therefore using Kernel trick, we can solve for optimal $\hat{\beta}$ instead of $\hat{\mathbf{w}}$ instead:

$$\begin{aligned}
\hat{\beta} &= \arg\min_{\beta} \quad \frac{1}{2}\sum_{i=1}^{N}\left(y_i - \sum_{j=1}^{N}\beta_j\ K(x_i,x_j)\right)^2 + \frac{\lambda}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\beta_i\beta_j\ K(x_i,x_j) \\
&= \arg\min_{\beta} \quad \frac{1}{2}\left(\beta^T K^T K\beta - 2\beta^T K^T y + y^T y\right) + \frac{\lambda}{2}\beta^T K\beta
\end{aligned}$$

We then find the closed form solution by setting the gradient equal to zero:

$$\begin{aligned}
\frac{\partial \frac{1}{2}\left(\beta^T K^T K\beta - 2\beta^T K^T y + y^T y\right) + \frac{\lambda}{2}\beta^T K\beta}{\partial\beta} &= 0 \\
\frac{\partial \frac{1}{2}\left(\beta^T K^T K\beta - 2\beta^T K^T y + y^T y\right)}{\partial\beta} + \frac{\partial \frac{\lambda}{2}\beta^T K\beta}{\partial\beta} &= 0 \\
K^T K\beta - K^T y + \lambda K^T I\beta &= 0 \\
K^T\left((K+\lambda I)\beta - y\right) &= 0 \\
\hat{\beta} &= (K+\lambda I)^{-1}y
\end{aligned}$$

By using the kernel trick on the regularized linear regression model, we can now solve for nonlinear regression problems.

Appendix

A1. Question One using Softmax

For binary classification problem, the task is to predict the likelihood of an example belonging to class one. In the lecture notes, logistic regression is used for such problems. On the other hand, for multi-class classification problem, the task is to predict the likelihood of an example belonging to each class. Therefore, the Softmax regression, which is a generalization of logistic regression, is more commonly used to obtain the normalized probabilities of the scores that sum to 1.

Given N training examples, where input features are $x_i \in \mathbb{R}^M$. For ease of notation, we assume there are C class $\{1, 2, 3..., C\}$ (index starts from 1 instead of 0 in lecture notes). The softmax function can be written as:

$$\hat{y_i} = P(\hat{y_i} = c|x_i; w^{(i)}) = \frac{\exp\left(w^{(i)^T} x_i\right)}{\sum_{j=1}^{C} \exp\left(w^{(j)^T} x_i\right)} \tag{14}$$

We can define the cross entropy loss function as follows:

$$E(w) = -\sum_{k=1}^{C} y_k log(\hat{y_k}) \tag{15}$$

The total loss function to minimize is:

$$J(w) = -\sum_{i=1}^{N} \left[-\sum_{k=1}^{C} y_k log(\hat{y_k}) \right] + \frac{\lambda}{2} \|\mathbf{w}\|_2^2 \tag{16}$$

Based on Gradient Descent update rule (where α is the learning rate), we need to compute the 1st order derivative of the loss function w.r.t the weights:

$$\begin{aligned} w_{t+1} &= w_t - \alpha \frac{\partial J(w)}{\partial w} \\ w_{t+1} &= w_t - \alpha \left(-\sum_{i=1}^{N} \frac{\partial E(w)}{\partial w} + \frac{\partial \frac{\lambda}{2} \|\mathbf{w}\|_2^2}{\partial w} \right) \end{aligned} \tag{17}$$

To derive this gradient $\frac{\partial E(w)}{\partial w}$, we can use the Chain Rule:

$$\frac{\partial E(w)}{\partial w} = \frac{\partial E(w)}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial z} \times \frac{\partial z}{\partial w}$$
$$\textit{where we let } z^{(i)} = w^{(i)^T} x_i$$

We will start by deriving the partial derivatives for the Softmax Function (Equation 14) w.r.t to z, $\frac{\partial \hat{y}}{\partial z}$:

$$\frac{\partial \hat{y_k}}{\partial z^{(i)}} = \frac{\partial}{\partial z^{(i)}} \left(\frac{\exp\left(w^{(k)^T} x_i\right)}{\sum_{j=1}^{C} \exp\left(w^{(j)^T} x_i\right)} \right)$$

Before applying Quotient Rule, we find the partial derivatives for $\frac{g(x)}{h(x)}$ separately:

$$\begin{aligned} &Let: \\ &\quad g(x) = \exp\left(w^{(k)^T} x_i\right) \\ &\quad h(x) = \sum_{j=1}^{C} \exp\left(w^{(j)^T} x_i\right) \end{aligned}$$

$$g'(x) = \frac{\partial g(x)}{\partial z^{(i)}} = \frac{\partial \exp\left(w^{(k)^T} x_i\right)}{\partial (w^{(i)^T} x_i)} = \begin{cases} \exp\left(w^{(i)^T} x_i\right) & i = k \\ 0 & i \neq k \end{cases}$$

$$h'(x) = \frac{\partial h(x)}{\partial z^{(i)}} = \frac{\partial \sum_{j=1}^{C} \exp\left(w^{(j)^T} x_i\right)}{\partial (w^{(i)^T} x_i)} = \exp\left(w^{(i)^T} x_i\right)$$

The Quotient Rule states that $f'(x) = \dfrac{g'(x)h(x) - h'(x)g(x)}{(h(x))^2}$:

Hence, for the case when $i = k$:

$$\begin{aligned}
\frac{\partial \hat{y}_k}{\partial z^{(i)}} &= \frac{\exp\left(w^{(i)^T} x_i\right)\left[\sum_{j=1}^{C} \exp\left(w^{(j)^T} x_i\right)\right] - \exp\left(w^{(i)^T} x_i\right)\exp\left(w^{(k)^T} x_i\right)}{\left(\sum_{j=1}^{C} \exp(w^{(j)^T} x_i)\right)^2} \\
&= \frac{\exp\left(w^{(i)^T} x_i\right)\left[\sum_{j=1}^{C} \exp\left(w^{(j)^T} x_i\right) - \exp\left(w^{(k)^T} x_i\right)\right]}{\left(\sum_{j=1}^{C} \exp(w^{(j)^T} x_i)\right)^2} \\
&= \frac{\exp\left(w^{(i)^T} x_i\right)}{\sum_{j=1}^{C} \exp(w^{(j)^T} x_i)} \; \frac{\sum_{j=1}^{C} \exp\left(w^{(j)^T} x_i\right) - \exp\left(w^{(k)^T} x_i\right)}{\sum_{j=1}^{C} \exp(w^{(j)^T} x_i)} \\
&= \frac{\exp\left(w^{(i)^T} x_i\right)}{\sum_{j=1}^{C} \exp(w^{(j)^T} x_i)} \left(\frac{\sum_{j=1}^{C} \exp\left(w^{(j)^T} x_i\right)}{\sum_{j=1}^{C} \exp(w^{(j)^T} x_i)} - \frac{\exp\left(w^{(k)^T} x_i\right)}{\sum_{j=1}^{C} \exp(w^{(j)^T} x_i)} \right) \\
&= \hat{y}_i \left(1 - \hat{y}_k\right) \\
&= \hat{y}_i \left(1 - \hat{y}_i\right)
\end{aligned} \tag{18}$$

Then, for the case when $i \neq k$:

$$\begin{aligned}
\frac{\partial \hat{y}_k}{\partial z^{(i)}} &= \frac{(0)\sum_{j=1}^{C} \exp\left(w^{(j)^T} x_i\right) - \exp\left(w^{(i)^T} x_i\right)\exp\left(w^{(k)^T} x_i\right)}{\left(\sum_{j=1}^{C} \exp(w^{(j)^T} x_i)\right)^2} \\
&= -\frac{\exp\left(w^{(i)^T} x_i\right)\exp\left(w^{(k)^T} x_i\right)}{\left(\sum_{j=1}^{C} \exp(w^{(j)^T} x_i)\right)^2} \\
&= -\frac{\exp\left(w^{(i)^T} x_i\right)}{\sum_{j=1}^{C} \exp(w^{(j)^T} x_i)} \; \frac{\exp\left(w^{(k)^T} x_i\right)}{\sum_{j=1}^{C} \exp(w^{(j)^T} x_i)} \\
&= -\hat{y}_i \left(\hat{y}_k\right)
\end{aligned} \tag{19}$$

Therefore with Equations 18 and 19, we can derive the derivatives for Softmax, $\dfrac{\partial \hat{y}}{\partial z^{(i)}}$ to be:

$$\frac{\partial \hat{y}_j}{\partial z^{(i)}} = \begin{cases} \hat{y}_i \left(1 - \hat{y}_i\right) & i = k \\ -\hat{y}_i \left(\hat{y}_k\right) & i \neq k \end{cases}$$

We can now compute the derivatives for Cross Entropy Loss:

$$
\begin{aligned}
\frac{\partial E(w)}{\partial z^{(i)}} &= \frac{\partial}{\partial z^{(i)}}\Big[-\sum_{k=1} y_k \ \log \hat{y}_k\Big] \\
&= -\sum_{k=1} y_k \left[\frac{\partial \log \hat{y}_k}{\partial z^{(i)}}\right] \\
&= -\sum_{k=1} y_k \left[\frac{\partial \log \hat{y}_k}{\partial \hat{y}_k} \times \frac{\partial \hat{y}_k}{\partial z^{(i)}}\right] \\
&= -\sum_{k=1} \left[\frac{y_k}{\hat{y}_k} \times \frac{\partial \hat{y}_k}{\partial z^{(i)}}\right] \\
&= -\left[\frac{y_i}{\hat{y}_i}\ \frac{\partial \hat{y}_i}{\partial z^{(i)}} + \sum_{k=1,k\neq i} \frac{y_k}{\hat{y}_k}\frac{\partial \hat{y}_k}{\partial z^{(i)}}\right] \\
&= -\left[\frac{y_i}{\hat{y}_i}\ \hat{y}_i(1-\hat{y}_i) \ + \sum_{k=1,k\neq i} \frac{y_k}{\hat{y}_k}\ (-\hat{y}_k\hat{y}_i)\right] \\
&= -\left[y_i\ (1-\hat{y}_i) \ - \sum_{k=1,k\neq i} y_k\hat{y}_i\right] \\
&= -y_i + y_i\hat{y}_i + \sum_{k=1,k\neq i} y_k\hat{y}_i \\
&= -y_i + \hat{y}_i\big(y_i + \sum_{k=1,k\neq i} y_k\big) \\
&= -y_i + \hat{y}_i\big(\sum_{k=1} y_k\big) \\
&= -y_i + \hat{y}_i\big(1\big) \\
&= \hat{y}_i \ - \ y_i
\end{aligned}
$$

The derivatives of $z^{(i)}$ w.r.t to the weights $w^{(i)}$ is:

$$
\begin{aligned}
\frac{\partial z^{(i)}}{\partial w^{(i)}} &= \frac{\partial ({w^{(i)}}^T x_i)}{\partial w^{(i)}} \\
&= x_i
\end{aligned}
$$

Hence,

$$
\begin{aligned}
\frac{\partial E(w)}{\partial w^{(i)}} &= \frac{\partial E(w)}{\partial z^{(i)}} \times \frac{\partial z^{(i)}}{\partial w^{(i)}} \\
&= (\hat{y}_i \ - \ y_i)\, x_i \qquad (20)
\end{aligned}
$$

Sub Equations 12 and 20 into Gradient Descent rule from Equation 17:

$$
\begin{aligned}
w_{t+1} &= w_t - \alpha\left(-\sum_{i=1}^{N} \frac{\partial E(w)}{\partial w} + \frac{\partial \frac{\lambda}{2}\|\mathbf{w}\|_2^2}{\partial w}\right) \\
w_{t+1} &= w_t + \alpha\left(\sum_{i=1}^{N} (\hat{y}_i \ - \ y_i)\, x_i - \lambda w\right) \qquad (21)
\end{aligned}
$$

From Equations 13 and 21, we can see that the gradient update rules are the same for both.