ISYE 6740 Homework 5
Fall 2020

Total 100 points + 10 bonus points.

Farshad Rafiei

1. **SVM.** (45 points)

   (a) (5 points) Explain why can we set the margin $c = 1$ to derive the SVM formulation?
   Answer: Solving a classification problem using SVM requires us to solve the following optimization problem:

   $$\max_{w,b} \frac{c}{||w||} \text{ subject to } y^i(w^T x^i + b) \geq c \tag{1}$$

   Magnitude of c is just a scale factor and does not change the relative goodness of different classifiers. Therefore, we can set c to any value including $c = 1$.

   (b) (10 points) Using Lagrangian dual formulation, show that the weight vector can be represented as

   $$w = \sum_{i=1}^{n} \alpha_i y_i x_i.$$

   where $\alpha_i \geq 0$ are the dual variables. What does this imply in terms of how to relate data to w?
   Answer: Lagrangian function is as following:

   $$L(w, b, \alpha) = \frac{1}{2} w^T w + \sum_{i=1}^{n} \alpha_i (1 - y^i(w^T x^i + b)) \tag{2}$$

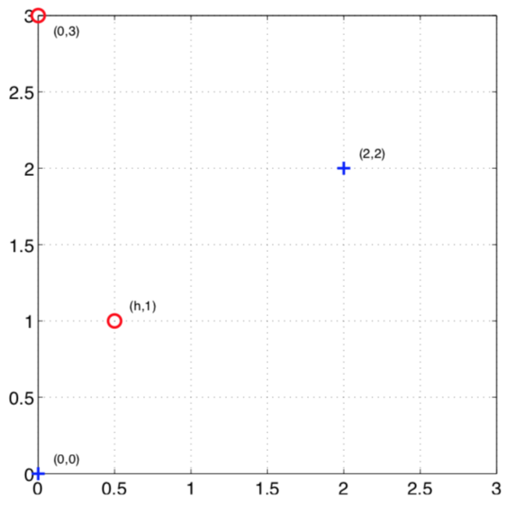
   To find the optimal weight vector w, we have to take the first derivative of above Lagrangian function with respect to w and set it to zero.

   $$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{n} \alpha_i y^i x^i = 0 \implies \boxed{w = \sum_{i=1}^{n} \alpha_i y^i x^i} \tag{3}$$

   For data points with $1 - y^i(w^T x^i + b) < 0$, $\alpha_i = 0$. But for rest of the data points in which $1 - y^i(w^T x^i + b) > 0$, we have to set $\alpha_i \geq 0$ to penalize for wrong classification. Therefore, weight vector w is the linear combination of subset of data points.

   (c) (10 points) Explain why only the data points on the "margin" will contribute to the sum above, i.e., playing a role in defining w.
   Answer: Data points with nonzero α_i are called support vectors. These support vectors are the ones which are close to the separating line (or generally hyperplane) and could potentially have $1 - y^i(w^T x^i + b) > 0$. Data points beyond the margin will not contribute to the weight vector since the criterion for them are automatically met in case of accounting for marginal data points.

(d) (20 points) Suppose we only have four training examples in two dimensions as shown in Fig. The positive samples at $x_1 = (0,0)$, $x_2 = (2,2)$ and negative samples at $x_3 = (h,1)$ and $x_4 = (0,3)$.



i. (10 points) For what range of parameter $h > 0$, the training points are still linearly separable?
Answer: We can find the equation of the line which passes through blue lines. That is $y = x$. For the two classes illustrated in figure, to be linearly separable, red points need to fall on the left side of the line $y = x$, since point $(0,3)$ is already on the left side of this line. Therefore, for every point in class "red points", we should have: $y > x$. Hence: $\boxed{h < 1}$

ii. (10 points) Does the orientation of the maximum margin decision boundary change as h changes, when the points are separable?
Answer: No, because x1, x2, x3 are the closest data points to the margin and they remain the support vectors.

2. **Multi-class classification for MNIST data set, comparison.** (55 points)

This question is to compare different classifiers and their performance for multi-class classifications on the complete MNIST dataset at `http://yann.lecun.com/exdb/mnist/`. You can find the data file mnist_10digits.mat in the homework folder. The MNIST database of handwritten digits has a training set of 60,000 examples, and a test set of 10,000 examples. We will compare **KNN, logistic regression, SVM, kernel SVM, and neural networks**. We suggest to use Scikit-learn, which is a commonly-used and powerful Python library with various machine learning tools. But you can also use other similar libraries in other programming languages of your choice to perform the tasks. Below are some tips.

- We suggest you to "standardize" the features before training the classifiers, by dividing the values of the features by 255 (thus map the range of the features from [0, 255] to [0, 1]).
- You may adjust the number of neighbors K used in KNN to have a reasonable result (you may use cross validation but it is not required; any reasonable tuning to get good result is acceptable).
- You may use a neural networks function sklearn.neural_network with hidden_layer_sizes = (20, 10).
- For kernel SVM, you may use radial basis function kernel, and a heuristic called "median trick": choose the parameter of the kernel $K(x, x') = \exp\{-\|x - x'\|^2/(2\sigma^2)\}$. Choose the bandwidth as $\sigma = \sqrt{M/2}$ where $M =$ the median of $\{\|x^i - x^j\|^2, 1 \leq i, j \leq m', i \neq j\}$ for pairs of training

samples. Here you can randomly choose $m' = 1000$ samples from training data to use for the "median trick"[1].

- For KNN and SVM, you can randomly downsample the training data to size $m = 5000$, to improve computation efficiency.

Train the classifiers on training dataset and evaluate on the test dataset.

(a) (50 points) Report confusion matrix, precision, recall, and F-1 score for each of the classifiers. For precision, recall, and F-1 score of each classifier, we will need to report these for each of the digits. So you can create a table for this. For this question, each of the 5 classifier, **KNN, logistic regression, SVM, kernel SVM, and neural networks**, accounts for 10 points.
Answer: First, we report the classification performance using KNN classifier. To find the optimal k, we used grid search cross validation in Scikit-Learn. The highest classification accuracy was found for $k = 4$ using training dataset with 10 fold cross validation. Performance of KNN classifier with $k = 4$ is summarized in Table 1 and Figure 1. **Overall accuracy is** 97%**.**

Table 1: Performance of KNN ($k = 4$) classifier on MNIST dataset.

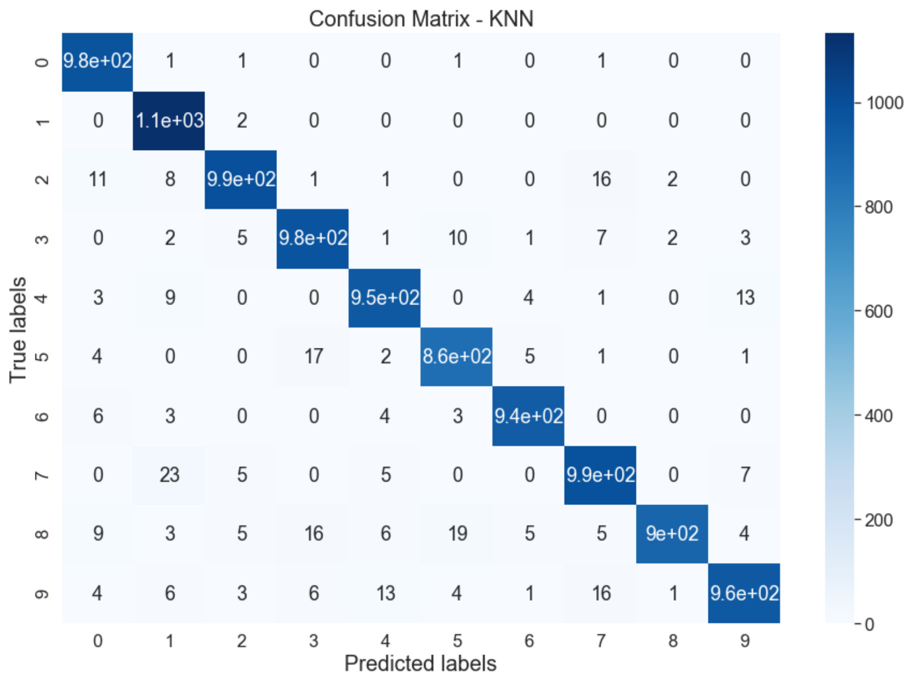
Digit	precision	recall	f1-score
0	0.96	1.00	0.98
1	0.95	1.00	0.98
2	0.98	0.96	0.97
3	0.96	0.97	0.97
4	0.97	0.97	0.97
5	0.96	0.97	0.96
6	0.98	0.98	0.98
7	0.95	0.96	0.96
8	0.99	0.93	0.96
9	0.97	0.95	0.96

Second classifier is Logistic Regression. Table 2 and Figure 2 show the classification performance for a logistic regression classifier based on $\ell 2$ regularization and regularization strength of $c = 1$. **Overall accuracy is** 93%**.**

Table 2: Performance of Logistic Regression classifier on MNIST dataset.

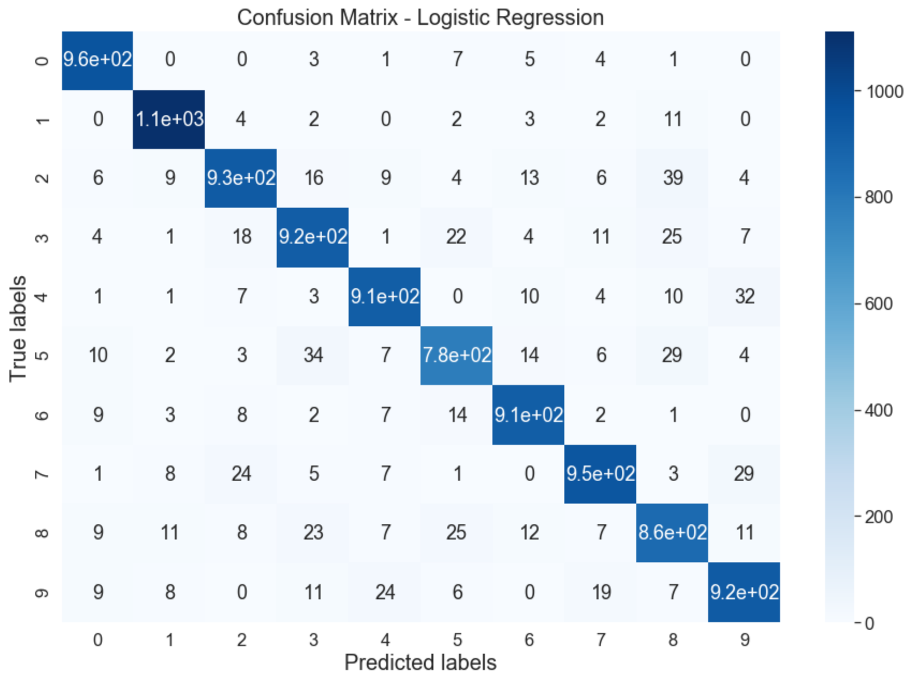
Digit	precision	recall	f1-score
0	0.95	0.98	0.96
1	0.96	0.98	0.97
2	0.93	0.90	0.91
3	0.90	0.91	0.91
4	0.94	0.93	0.93
5	0.91	0.88	0.89
6	0.94	0.95	0.94
7	0.94	0.92	0.93
8	0.87	0.88	0.88
9	0.91	0.92	0.92

Third classifier is the Linear SVM. We used linear kernel with regularization parameter of $c = 1$. Results are illustrated in Table 3 and Figure 3. **Overall accuracy is** 94%**.**



	0	1	2	3	4	5	6	7	8	9
0	9.8e+02	1	1	0	0	1	0	1	0	0
1	0	1.1e+03	2	0	0	0	0	0	0	0
2	11	8	9.9e+02	1	1	0	0	16	2	0
3	0	2	5	9.8e+02	1	10	1	7	2	3
4	3	9	0	0	9.5e+02	0	4	1	0	13
5	4	0	0	17	2	8.6e+02	5	1	0	1
6	6	3	0	0	4	3	9.4e+02	0	0	0
7	0	23	5	0	5	0	0	9.9e+02	0	7
8	9	3	5	16	6	19	5	5	9e+02	4
9	4	6	3	6	13	4	1	16	1	9.6e+02

Figure 1: **Confusion matrix for classification performance on MNIST daatset using KNN** ($k = 4$) **classifier**.



	0	1	2	3	4	5	6	7	8	9
0	9.6e+02	0	0	3	1	7	5	4	1	0
1	0	1.1e+03	4	2	0	2	3	2	11	0
2	6	9	9.3e+02	16	9	4	13	6	39	4
3	4	1	18	9.2e+02	1	22	4	11	25	7
4	1	1	7	3	9.1e+02	0	10	4	10	32
5	10	2	3	34	7	7.8e+02	14	6	29	4
6	9	3	8	2	7	14	9.1e+02	2	1	0
7	1	8	24	5	7	1	0	9.5e+02	3	29
8	9	11	8	23	7	25	12	7	8.6e+02	11
9	9	8	0	11	24	6	0	19	7	9.2e+02

Figure 2: **Confusion matrix for classification performance on MNIST daatset using Logistic Regression classifier**.

Table 3: Performance of Linear SVM classifier on MNIST dataset.

Digit	precision	recall	f1-score
0	0.95	0.98	0.96
1	0.97	0.99	0.98
2	0.93	0.94	0.93
3	0.91	0.94	0.92
4	0.94	0.96	0.95
5	0.91	0.90	0.91
6	0.96	0.95	0.95
7	0.95	0.93	0.94
8	0.94	0.90	0.92
9	0.95	0.91	0.93

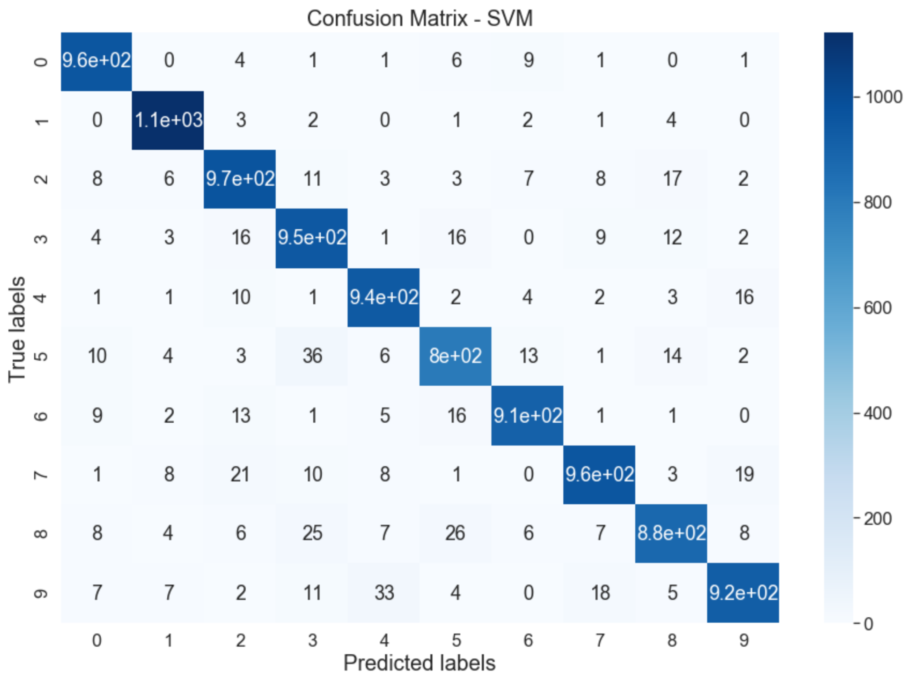


Figure 3: **Confusion matrix for classification performance on MNIST daatset using Linear SVM**.

Forth classifier we used is kernel (RBF) SVM. Similar to linear SVM, we used regularization parameter of $c = 1$. Results are provided in Table 4 and Figure 4. **Overall accuracy is** 98%**.**

Finally, we recruited neural networks to perform classification on MNIST dataset. We used two hidden layers which first layer consisted of 20 neurons and second hidden layer had 10 neurons. Regularization parameter was set to $\alpha = 0.0001$ and learning rate was set to 0.05 with stochastic gradient descent (SGD) being the solver. Results are shown in Table 5 and Figure 5. **Overall accuracy is** 96%**.**

(b) (5 points) Comment on the performance of the classifier and give your explanation why some of them perform better than the others.

Answer: Performance of all these classifiers for MNIST dataset is reasonable and there is no

[1]Garreau, Damien, Wittawat Jitkrittum, and Motonobu Kanagawa. "Large sample analysis of the median heuristic." arXiv preprint arXiv:1707.07269 (2017).

Table 4: Performance of Kernel (RBF) SVM classifier on MNIST dataset.

Digit	precision	recall	f1-score
0	0.98	0.99	0.99
1	0.99	0.99	0.99
2	0.98	0.97	0.98
3	0.97	0.99	0.98
4	0.98	0.98	0.98
5	0.99	0.98	0.98
6	0.99	0.99	0.99
7	0.98	0.97	0.97
8	0.97	0.98	0.97
9	0.97	0.96	0.97

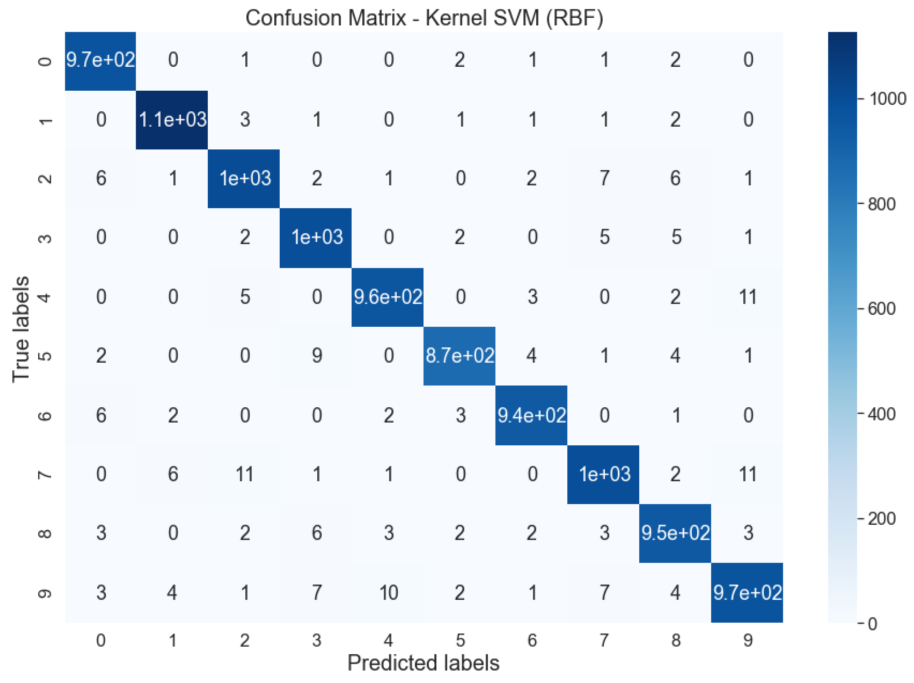
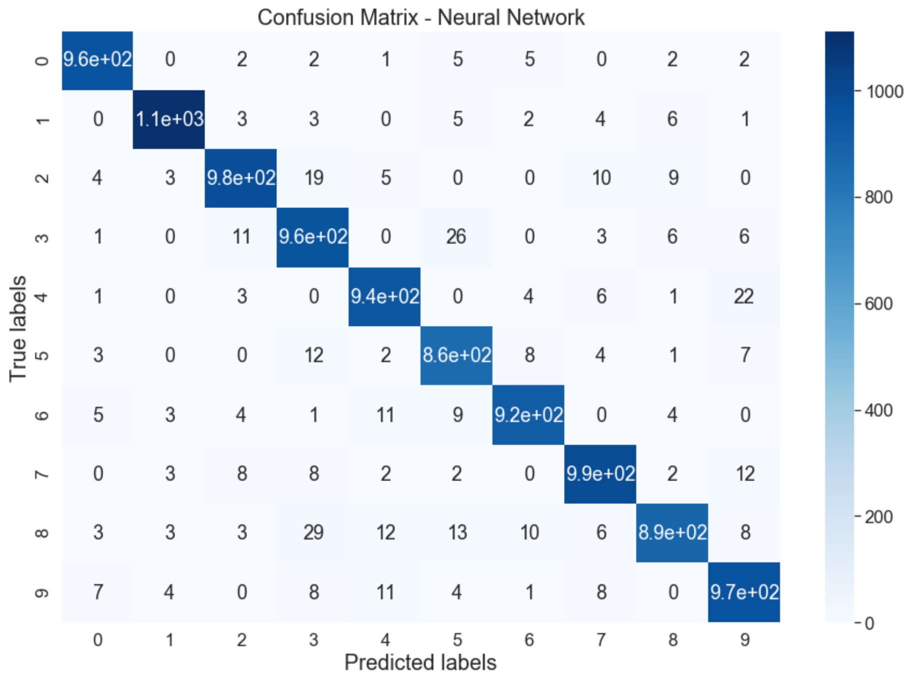


Figure 4: **Confusion matrix for classification performance on MNIST daatset using Kernel (RBF) SVM**.

significant difference between them. However, there are some small differences. KNN performs very well on this dataset. In fact, it achieved the highest accuracy but the running time was the highest compared to rest of the classifiers. RBF Svm performed as well as KNN but the running time for it was relatively high too.
Logistic regression and linear SVM perform equally well and the training time for them is less than KNN and kernel SVM. However, these models experience difficulty classifying certain digits from others. For example, when it comes to recognizing 2 and 7, these two models tend to mix these digits up and show low accuracy in confusion matrix.
Neural network was the best among all classifiers. It takes only 10 iterations for it to achieve a reasonable performance. In this problem, we used the hidden layer information provided in assignment and achieved 96% accuracy using neural networks. However, we realized that alternative

Table 5: Performance of Neural Network on classification of MNIST dataset.

Digit	precision	recall	f1-score
0	0.98	0.98	0.98
1	0.99	0.98	0.98
2	0.97	0.95	0.96
3	0.92	0.95	0.93
4	0.96	0.96	0.96
5	0.93	0.96	0.94
6	0.97	0.96	0.96
7	0.96	0.96	0.96
8	0.97	0.91	0.94
9	0.94	0.96	0.95



True \ Predicted	0	1	2	3	4	5	6	7	8	9
0	9.6e+02	0	2	2	1	5	5	0	2	2
1	0	1.1e+03	3	3	0	5	2	4	6	1
2	4	3	9.8e+02	19	5	0	0	10	9	0
3	1	0	11	9.6e+02	0	26	0	3	6	6
4	1	0	3	0	9.4e+02	0	4	6	1	22
5	3	0	0	12	2	8.6e+02	8	4	1	7
6	5	3	4	1	11	9	9.2e+02	0	4	0
7	0	3	8	8	2	2	0	9.9e+02	2	12
8	3	3	3	29	12	13	10	6	8.9e+02	8
9	7	4	0	8	11	4	1	8	0	9.7e+02

Figure 5: **Confusion matrix for classification performance on MNIST daatset using Neural Networks**.

models with only one hidden unit which contains 50 nodes can achieve more than 98% accuracy in classification of MNIST dataset.

3. **Neural networks.** (Bonus: 10 points)

   (a) (2 points) Consider a neural networks for a binary classification using sigmoid function for each unit. If the network has no hidden layer, explain why the model is equivalent to logistic regression.
   Answer: The intuition behind logistic regression is that the to learn parameters θ such that the weighted summation of a single data point including different features adds up to a single number. Once it's done, then that single number serves as an input to the sigmoid function and logistic regression decides which category that single number belongs to. The neural network with single sigmoid function at the output acts identical to this procedure. All inputs are summed together

and go through a sigmoid function which makes the decision based on the number received. All it needs to do is to learn the weights on edges that connects input nodes to the output node.

(b) (8 points) Consider a simple two-layer network in the lecture slides. Given m training data (x^i, y^i), $i = 1, \ldots, m$, the cost function used to training the neural networks

$$\ell(w, \alpha, \beta) = \sum_{i=1}^{m} (y^i - \sigma(w^T z^i))^2$$

where $\sigma(x) = 1/(1 + e^{-x})$ is the sigmoid function, z^i is a two-dimensional vector such that $z_1^i = \sigma(\alpha^T x^i)$, and $z_2^i = \sigma(\beta^T x^i)$. Show the that the gradient is given by

$$\frac{\partial \ell(w, \alpha, \beta)}{\partial w} = -\sum_{i=1}^{m} 2(y^i - \sigma(u^i))\sigma(u^i)(1 - \sigma(u^i))z^i,$$

where $u^i = w^T z^i$. Also find the gradient of $\ell(w, \alpha, \beta)$ with respect to α and β and write down their expression.

Answer:

$$\begin{aligned} \frac{\partial \ell(w, \alpha, \beta)}{\partial w} &= \frac{\partial}{\partial w} \sum_{i=1}^{m} (y^i - \sigma(u^i))^2 \\ &= -\sum_{i=1}^{m} 2(y^i - \sigma(u^i))\{\frac{\partial}{\partial w}\sigma(u^i)\} \end{aligned} \tag{4}$$

$$\begin{aligned} \frac{\partial}{\partial w}\sigma(u^i) &= \frac{\partial}{\partial w} \frac{1}{1 + e^{-u^i}} \\ &= \frac{\partial}{\partial w} \frac{1}{1 + e^{-w^T z^i}} \\ &= \frac{e^{-w^T z^i}}{(1 + e^{-w^T z^i})^2} z^i \\ &= \frac{e^{-u^i}}{(1 + e^{-u^i})^2} z^i \\ &= \frac{e^{-u^i}}{1 + e^{-u^i}} \sigma(u^i) z^i \\ &= (1 - \sigma(u^i))\sigma(u^i) z^i \end{aligned} \tag{5}$$

By plugging equation (5) into equation (4), we have:

$$\frac{\partial \ell(w, \alpha, \beta)}{\partial w} = -\sum_{i=1}^{m} 2(y^i - \sigma(u^i))\sigma(u^i)(1 - \sigma(u^i))z^i,$$

Now, let's find the gradient of $\ell(w, \alpha, \beta)$ with respect to α:

$$\frac{\partial \ell(w, \alpha, \beta)}{\partial \alpha} = \frac{\partial \ell(w, \alpha, \beta)}{\partial z_1^i} \frac{\partial z_1^i}{\partial \alpha} \tag{6}$$

The first part of above equation, $\frac{\partial \ell(w,\alpha,\beta)}{\partial z_1^i}$, is similar to the gradient of ℓ with respect to w. Therefore:

$$\frac{\partial \ell(w,\alpha,\beta)}{\partial z_1^i} = -\sum_{i=1}^{m} 2(y^i - \sigma(u^i))\sigma(u^i)(1-\sigma(u^i))w_1 \tag{7}$$

Where, w_1 is the weight corresponding to z_1^i. Now, we need to compute the second part of the equation in (6), which is $\frac{\partial z_1^i}{\partial \alpha}$.

$$\begin{aligned} \frac{z_1^i}{\partial \alpha} &= \frac{\partial}{\partial \alpha}\sigma(\alpha^T x^i) \\ &= \sigma(\alpha^T x^i)(1-\sigma(\alpha^T x^i))x^i \end{aligned} \tag{8}$$

By plugging equations (7) and (8) into equation (6), we have:

$$\frac{\partial \ell(w,\alpha,\beta)}{\partial \alpha} = -\sum_{i=1}^{m} 2(y^i - \sigma(u^i))\sigma(u^i)(1-\sigma(u^i))w_1\sigma(\alpha^T x^i)(1-\sigma(\alpha^T x^i))x^i \tag{9}$$

Similarly,

$$\frac{\partial \ell(w,\alpha,\beta)}{\partial \beta} = -\sum_{i=1}^{m} 2(y^i - \sigma(u^i))\sigma(u^i)(1-\sigma(u^i))w_2\sigma(\beta^T x^i)(1-\sigma(\beta^T x^i))x^i \tag{10}$$