Introduction to ML (CS771), 2024-2025-Sem-I Quiz 2. September 7, 2024		Total Marks	25
		Duration	45 minutes
Name		Roll No.	

Instructions:

1.	Clearly write your name (in block letters) and roll number in the provided boxes above.
2.	Write your final answers concisely in the provided space. You may use blue/black pen.
3.	We won't be able to provide clarifications during the quiz. If any aspect of some question appears ambiguous/unclear to you, please state your assumption(s) and answer accordingly.

Question 1: Write **T** or **F** for True/False in the box next to each question given below, with a brief (1-2 sentences at most) explanation in the provided space in the box below the question. Marks will be awarded only when the answer (T/F) and explanation both are correct. (**3 x 2 = 6 marks**)

1.1	In any iteration $t = 1,2,\dots,T$ of gradient descent (GD) for linear regression, the gradient expression is more highly influenced by those training examples $(\boldsymbol{x}_n, y_n)$ on which the current $\boldsymbol{w}^{(t)}$ has a small error (i.e., difference between y_n and $\boldsymbol{w}^{(t)\top}\boldsymbol{x}_n$).	**F**

The gradient expression has a summation over all the training examples and each of the terms in the summation is of the form $(y_n - \boldsymbol{w}^{(t)\top}\boldsymbol{x}_n)\boldsymbol{x}_n$. So training examples on which the current model has a large error will influence the gradient more.

1.2	The absolute value loss function $\lvert y_n - \boldsymbol{w}^\top \boldsymbol{x}_n \rvert$ for linear regression cannot be optimized using first-order optimality to get a closed form solution for the weight vector $\boldsymbol{w}$	**T**

Yes, because the absolute loss function is not differentiable.

1.3	The Perceptron loss function defined as $\max\{0, -y_n \boldsymbol{w}^\top \boldsymbol{x}_n\}$ is not differentiable but the Hinge loss function defined as $\max\{0, 1 - y_n \boldsymbol{w}^\top \boldsymbol{x}_n\}$ is differentiable.	**F**

Both have the same shape except being a horizontally shifted version of each other. Both are non-differentiable at the point the loss' value start becoming nonzero.

Question 2: Answer the following questions concisely in the space provided below the question.

2.1	Mention two advantages of Newton's method for optimization as compared to gradient descent, and also one disadvantage of the former. **(4 marks)**

Two advantages: (1) Converges in fewer iterations, and sometimes may be faster in terms of overall time too, (2) Does not require specifying a learning rate (Hessian defines the learning rate)

One disadvantage: It can be computationally more expensive especially when input dimensionality (D) is large. Recall that we have to compute and invert Hessian matrix in each iteration.

2.2	Given the confusion matrix for the test data in a multi-class classification problem, can you compute the accuracy? If yes, how? If not, why not? **(3 marks)**

Yes. Sum the diagonal entries (note that each entry denotes how many test inputs of a particular class got their label predicted corrected) and divide the sum by the total number of examples.

2.3	The soft-margin SVM problem for binary classification minimizes the following loss function: $L(\boldsymbol{w}, b) = \frac{\|\boldsymbol{w}\|^2}{2} + C\sum_{n=1}^{N}\xi_n$ where $\xi_n > 0$ denotes the slack on the n^{th} training example. What would be the effect of using a very-very large value of C? Would the model tend to overfit or underfit? Also, what about the margin of the classifier? Will we get a large margin or small margin? Briefly justify your answer. **(4 marks)**

Note that $\frac{\|\boldsymbol{w}\|^2}{2}$ is the basically the inverse of the margin value, and the sum of slacks terms $\sum_{n=1}^{N}\xi_n$ is basically the Hinge loss (basically the training error) of SVM.

Increasing the "trade-off" hyperparameter C to a very-very large value will cause the optimization to give significantly more importance to the Hinge loss term as compared to the margin term, thereby pushing the Hinge loss (training error) to become very small (which could potentially lead result in overfitting), at the cost of not being left with much of the margin on either side of the hyperplane.

2.4	Assuming multi-class classification given N training examples and a total of C classes, write down the expression of the multi-class cross-entropy loss function, clearly and succinctly defining the terms/notation involved in the expression, and briefly explain why this is a suitable loss function for multi-class classification problems. **(4 marks)**

The multi-class cross-entropy loss is $L(\boldsymbol{W}) = -\sum_{n=1}^{N}\sum_{i=1}^{C} y_{n,i}\, log\, \mu_{n,i}$

In the above expression
$y_{n,i} = 1$ if the true label is i, and zero otherwise.
$\mu_{n,i} \propto \exp(w_i^\top \boldsymbol{x}_n)$ denotes the model's predicted probability that the true label is i

The loss function makes sense because its minimization will encourage $\mu_{n,i}$ to be high (ideally very close to 1) when $y_{n,i} = 1$, and $\mu_{n,i}$ to be low (ideally very close to 0) when $y_{n,i} = 0$.

2.5	Given a linear regression problem with non-negativity constraints on each entry of the weight vector $\boldsymbol{w}$, which of these two approaches would you prefer and why: (1) Projected Gradient Descent, and (2) Lagrangian based Optimization? **(4 marks)**

In this problem, PGD will be preferable because the projection operator is really-really simple (in each iteration of standard GD, simply setting the each negative entry of the weight vector $\boldsymbol{w}$ to zero). Lagrangian based optimization will have a relatively much higher overhead because of the introduction of additional variables (one for each of the D constraints here) and overall complicated objective resulting out of it.