Due Date: February 28th 2025, 23:00

Instructions

- *For all questions that are not graded only on the answer, show your work! Any problem without work shown will get no marks regardless of the correctness of the final answer.*
- *Please try to use a document preparation system such as LaTeX.* ***If you write your answers by hand, note that you risk losing marks if your writing is illegible without any possibility of regrade, at the discretion of the grader.***
- *Submit your answers electronically via the course GradeScope.* ***Incorrectly assigned answers can be given 0 automatically at the discretion of the grader.*** *To assign answers properly, please*
  - *Make sure that the top of the first assigned page is the question being graded.*
  - *Do not include any part of answer to any other questions within the assigned pages.*
  - *Assigned pages need to be placed in order.*
  - *For questions with multiple parts, the answers should be written in order of the parts within the question.*
- ***Questions requiring written responses should be short and concise when necessary.*** *Unnecessary wordiness will be penalized at the grader's discretion.*
- *Please sign the agreement below.*
- *It is your responsibility to follow updates to the assignment after release. All changes will be visible on Overleaf and Piazza.*
- *Any questions should be directed towards the TA for this assignment (theoretical part):* ***Jerry Huang****.*

I acknowledge I have read the above instructions and will abide by them throughout this assignment. I further acknowledge that any assignment submitted without the following form completed will result in no marks being given for this portion of the assignment.

Signature: _____

Name: **Félix Wilhelmy**

UdeM Student ID: **20333575**

Question 1

Question 1.1.a

False

Question 1.1.b

True

Question 1.1.c

True

Question 1.2.a

$$f(x) \quad \leq \quad \frac{\boldsymbol{b}-\boldsymbol{x}}{b-a} f(a) + \frac{\boldsymbol{x}-\boldsymbol{a}}{b-a} f(b)$$

Question 1.2.b

$$\int_0^1 f(x+\lambda(y-x))d\lambda \leq \quad \underline{\frac{1}{2}(f(x)+f(y))}$$

Question 1.3.a

False

Question 1.3.b

True

Question 1.4.a

2

Question 1.4.b

$\frac{1}{4}$

Question 1.5

False

Question 2

Question 2.1

Since $f(\mathbf{x})$ is a quadratic function (and hence convex), its unique global minimum can be found by setting its gradient equal to zero.

The gradient of $f(\mathbf{x})$ with respect to $\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$ is

$$\nabla f(\mathbf{x}) = \begin{pmatrix} \dfrac{\partial f}{\partial x_1} \\ \dfrac{\partial f}{\partial x_2} \end{pmatrix} = \begin{pmatrix} x_1 \\ c\, x_2 \end{pmatrix}.$$

Since $c > 0$, we get the unique optimal point (or global minimum) of $f(\mathbf{x})$

$$\mathbf{x}^* = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Question 2.2

Let's consider the objective function

$$f(\mathbf{x}) = \frac{\mathbf{1}}{\mathbf{2}} \left(\mathbf{x_1^2} + \mathbf{c}\, \mathbf{x_2^2} \right), \text{ with } \mathbf{c} > \mathbf{0}$$

The gradient of this function (as shown in the previous question) is

$$\nabla f(\mathbf{x}) = \begin{pmatrix} x_1 \\ c\, x_2 \end{pmatrix},$$

It's **Hessian** H is

$$H = \nabla^2 f(\mathbf{x}) = \begin{pmatrix} 1 & 0 \\ 0 & c \end{pmatrix}.$$

Newton's method updates the iterate $\mathbf{x^{(k)}}$ using the formula

$$\mathbf{x^{(k+1)}} = \mathbf{x^{(k)}} - \mathbf{H^{-1}}\nabla\mathbf{f}(\mathbf{x^{(k)}}).$$

The inverse of the Hessian H is

$$H^{-1} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{c} \end{pmatrix}.$$

Thus

$$\mathbf{x^{(k+1)}} = \mathbf{x^{(k)}} - \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{c} \end{pmatrix} \begin{pmatrix} x_1^{(k)} \\ c\, x_2^{(k)} \end{pmatrix}.$$

Performing the matrix multiplication, we obtain

$$\begin{pmatrix} 1 \cdot x_1^{(k)} + 0 \cdot (c\, x_2^{(k)}) \\ 0 \cdot x_1^{(k)} + \frac{1}{c} \cdot (c\, x_2^{(k)}) \end{pmatrix} = \begin{pmatrix} x_1^{(k)} \\ x_2^{(k)} \end{pmatrix}.$$

Therefore, the update rule simplifies to

$$\mathbf{x^{(k+1)}} = \mathbf{x^{(k)}} - \mathbf{x^{(k)}} = \mathbf{0}.$$

This shows that regardless of the starting point $\mathbf{x^{(0)}}$, Newton's method yields

$$\mathbf{x^{(1)}} = \mathbf{0},$$

Question 2.3

Recall the quadratic function

$$f(\mathbf{x}) = \frac{\mathbf{1}}{\mathbf{2}} \left(\mathbf{x_1^2} + \mathbf{c}\, \mathbf{x_2^2} \right),$$

with $c > 0$. In question 2.2, we derived that regardless of the starting point $\mathbf{x^{(0)}}$, the Newton's method yields

$$\mathbf{x^{(1)} = 0},$$

This means that for the given quadratic function, the Newton's method converges in a single iteration for any given point.

Question 2.4

When $c = 1$, our objective function becomes

$$f(\mathbf{x}) = \frac{\mathbf{1}}{\mathbf{2}} \left(\mathbf{x_1^2} + \mathbf{x_2^2} \right).$$

Its Hessian is the identity matrix

$$H = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

Since the Hessian is the identity matrix, the optimal learning rate will be 1, and the uniform curvature (condition number 1) will ensure rapid convergence.

Question 2.5

Consider the quadratic function and its Hessian

$$f(\mathbf{x}) = \frac{\mathbf{1}}{\mathbf{2}} \left(\mathbf{x_1^2} + \mathbf{c}\, \mathbf{x_2^2} \right), \text{ with } \mathbf{c} > \mathbf{0} \text{ and } \mathbf{H} = \begin{pmatrix} 1 & 0 \\ 0 & c \end{pmatrix}.$$

In gradient descent, to make sure we don't take too large a step (which might make us overshoot the minimum), we need the learning rate α to be smaller than a certain value. In order for the gradient descent to converge on a quadratic function, the step size must satisfy

$$0 < \alpha < \frac{2}{\lambda_{max}}.$$

Where λ_{max} is the maximum eigenvalue of the Hessian of our function. The eigenvalues of H are $\lambda_1 = 1$ and $\lambda_2 = c$. Thus we can advance that the gradient descent will converge for a learning rate that satisfies this condition

$$0 < \alpha < \frac{2}{\max\{1, c\}}.$$

This is true for any point and being on $\mathbf{x}^{(\mathbf{0})} = (\mathbf{1}, \mathbf{0})$ does not change that.

Question 3

Question 3.1

See Table [1]

Layer	Output Dimensions	Number of Parameters
Input	$32 \times 32 \times 3$	0
Conv(3, 12)	$32 \times 32 \times 12$	336
BatchNorm	$32 \times 32 \times 12$	24
ReLU	$32 \times 32 \times 12$	0
MaxPooling(2)	$16 \times 16 \times 12$	0
Conv(3, 8)	$16 \times 16 \times 8$	872
BatchNorm	$16 \times 16 \times 8$	16
ReLU	$16 \times 16 \times 8$	0
MaxPooling(2)	$8 \times 8 \times 8$	0
Reshape	$512 \times 1 \times 1$	0
FC(10)	$10 \times 1 \times 1$	5130

Table 1: Table to fill out.

Question 3.2

You can remove the bias from the two convolution layers because the subsequent batch normalisation removes the bias (offsets) added by each kernel during normalisation. Therefore, the bias are redundant parameters that could be removed from the model without affecting the final output.

If we count the bias for each convolution, we end up with a total of **20 parameters** (12 for the first convolution and 8 for the second) that are redundant and could be removed from the model.

Question 3.3

See Table [2]

Layer	Receptive Field
Conv(3,1) with stride **3**	6
MaxPooling(2) with stride 2	9
Conv(2,1) with stride 1	15
MaxPooling(3) with stride 3	27
Conv(2,1) with stride 1	45

Table 2: Table to fill out for part 3.

Question 4

Question 4.1.a

In this setting, we consider a binary classification problem where a true label $y \in \{+1, -1\}$ exists for every input $\mathbf{x}$, but we do not have access to y. Instead, we are provided with a confidence score defined as

$$c(\mathbf{x}) = \mathbf{p}(\mathbf{y} = +\mathbf{1} \mid \mathbf{x}) \neq \mathbf{0}$$

Since the sum of the probabilities of all outcomes must equal 1, we have

$$1 - c(\mathbf{x}) = \mathbf{p}(\mathbf{y} = -\mathbf{1} \mid \mathbf{x})$$

By applying the *Law of Total Expectation*, we can decompose the risk into an outer expectation over inputs $\mathbf{x}$ drawn from $p(\mathbf{x})$ and an inner expectation over labels y conditioned on $\mathbf{x}$. In the *discrete case*, this inner expectation can further be written as a summation over all possible values of $y \in \mathcal{Y}$

$$\mathcal{R}(g) = \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathbf{p}(\mathbf{x})}\Big[\mathop{\mathbb{E}}_{y\sim p(y|\mathbf{x})}\big[\ell\big(g(\mathbf{x}), \mathbf{y}\big)\big]\Big] = \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathbf{p}(\mathbf{x})}\Big[\sum_{\mathbf{y}\in\mathcal{Y}} \mathbf{p}(\mathbf{y} \mid \mathbf{x})\, \ell\big(\mathbf{g}(\mathbf{x}), \mathbf{y}\big)\Big]$$

In this setting and because y is binary, we obtain

$$\mathop{\mathbb{E}}_{y\sim p(y|\mathbf{x})}\big[\ell\big(g(\mathbf{x}), \mathbf{y}\big)\big] = \mathbf{c}(\mathbf{x})\, \ell\big(\mathbf{g}(\mathbf{x}), +\mathbf{1}\big) + \big(\mathbf{1} - \mathbf{c}(\mathbf{x})\big)\, \ell\big(\mathbf{g}(\mathbf{x}), -\mathbf{1}\big).$$

Thus, by substituting back into the expression for $\mathcal{R}(g)$, we obtain the risk in this setting

$$\mathcal{R}(g) = \mathop{\mathbb{E}}_{\mathbf{x},\mathbf{y}\sim\mathbf{p}(\mathcal{X},\mathcal{Y})}\big[c(\mathbf{x})\, \ell\big(\mathbf{g}(\mathbf{x}), +\mathbf{1}\big) + \big(\mathbf{1} - \mathbf{c}(\mathbf{x})\big)\, \ell\big(\mathbf{g}(\mathbf{x}), -\mathbf{1}\big)\big]\,.$$

Question 4.1.b

In this question, instead of the full distribution $p(\mathbf{x})$, we want to approximate the expectation with a finite training set

$$\mathcal{D}_{\text{tr}} = \{\mathbf{x_i}; \mathbf{c}(\mathbf{x_i})\}_{\mathbf{i=1}}^{\mathbf{N}}.$$

The *empirical risk* is defined as the average loss over the training set. Thus, the empirical risk is given by

$$\widehat{\mathcal{R}}(g) = \frac{1}{N}\sum_{i=1}^{N}\Big[c(\mathbf{x_i})\,\ell\big(\mathbf{g}(\mathbf{x_i}), +\mathbf{1}\big) + \Big(\mathbf{1} - \mathbf{c}(\mathbf{x_i})\Big)\,\ell\big(\mathbf{g}(\mathbf{x_i}), -\mathbf{1}\big)\Big].$$

Question 4.1.c

Recall that in our setting we do not have direct access to the true labels $y \in \{+1, -1\}$. Instead, for each input $\boldsymbol{x}$ we are given a confidence score

$$c(\boldsymbol{x}) = p(y = +1 \mid \boldsymbol{x}),$$

The training objective function proposed in the question is

$$L = \sum_{i=1}^{N}\Big[c(\boldsymbol{x}_i)\,\ell\big(g(\boldsymbol{x}_i), y_i\big) + \big(1 - c(\boldsymbol{x}_i)\big)\,\ell\big(-g(\boldsymbol{x}_i), y_i\big)\Big].$$

There are two issues with this proposed function

1. The expression uses the true label y_i. However, in our setting, y_i is not available; only the confidence scores $c(\boldsymbol{x}_i)$ can be observed. Therefore, the objective relies on unavailable information.
2. The second term uses the loss $\ell\big(-g(\boldsymbol{x}_i), y_i\big)$. Simply negating the output $g(\boldsymbol{x}_i)$ does not yield the correct loss for the negative class. This operation means that the objective will not accurately reflect the true risk.

Thus, it is not an appropriate loss function to minimize in this setting.

Question 4.2.a

Once again, let's consider a binary classification problem, but this time the data is unlabeled. Now, let's suppose that the training set is drawn from the following mixture distribution

$$p_{\text{tr}}(\mathbf{x}) = \theta\, \mathbf{p}_+(\mathbf{x}) + (\mathbf{1} - \theta)\, \mathbf{p}_-(\mathbf{x}),$$

where $\theta \in [0, 1]$ is a known constant representing the proportion of positive examples in the training data.

Here, $p_+(\mathbf{x})$ and $p_-(\mathbf{x})$ are class-conditional distributions that do not overlap

$$p_+(\mathbf{x}) = \mathbf{p}(\mathbf{x} \mid \mathbf{y} = +\mathbf{1}) \quad \text{and} \quad \mathbf{p}_-(\mathbf{x}) = \mathbf{p}(\mathbf{x} \mid \mathbf{y} = -\mathbf{1}),$$

We then define the expected loss of the classes as

$$e_+ = \mathbb{E}_{\mathbf{x}\sim\mathbf{p}_+}\left[\ell\big(g(\mathbf{x}), +\mathbf{1}\big)\right] \quad \text{and} \quad e_- = \mathbb{E}_{\mathbf{x}\sim\mathbf{p}_-}\left[\ell\big(g(\mathbf{x}), -\mathbf{1}\big)\right],$$

If the labels were available, the true risk of a predictor g under a loss function ℓ could be written as

$$R(g) = \theta\, e_+ + (1 - \theta)\, e_-.$$

However, since the training data is unlabeled and comes from the mixture $p_{\text{tr}}(\mathbf{x})$, we can only measure the overall error on the training set

$$R_{\text{tr}}(g) = \theta\, e_+ + (1 - \theta)\, e_-.$$

This expression is then a single equation with two unknowns e_+ and e_-. Without additional information, there is no way to uniquely determine e_+ and e_- from $R_{\text{tr}}(g)$ alone. Because of that, it is impossible to separately estimate the class-specific error rates based solely on the unlabeled mixture, making it impossible to accurately determine $R(g)$.

Question 4.2.b

Let's now suppose we have a second unlabeled training set drawn from a different mixture distribution θ' which is different from $\theta \neq \theta'$. This gives us a second equation such as

1. $R_{\text{tr}}(g) = \theta\, e_+ + (1-\theta)\, e_-$.
2. $R'_{\text{tr}}(g) = \theta'\, e_+ + (1-\theta')\, e_-$.

Since this system now contains two equations and two unknowns, we now have a simple solvable linear system of equations. Once the class-specific error rate e_+ and e_- are determined, we can compute the true risk as

$$R(g) = \theta\, e_+ + (1-\theta)\, e_-.$$

Question 4.3.a

Let's consider a modified risk estimator

$$\widetilde{R}(g) = \left|\widehat{R}(g) - \epsilon\right| + \epsilon,$$

where ϵ is a constant chosen as a baseline risk level.

My initial intuition on this new risk estimator $\widetilde{\mathcal{R}}(g)$ is that it will be punishing the predictor when it is too precise (or overfitting). We can confirm that by breaking this equation into two cases:

1) **When $\widehat{\mathcal{R}}(g) > \epsilon$, then**

$$|\widehat{\mathcal{R}}(g) - \epsilon| = \widehat{\mathcal{R}}(g) - \epsilon$$

$$\widetilde{\mathcal{R}}(g) = \widehat{\mathcal{R}}(g) - \epsilon + \epsilon = \widehat{\mathcal{R}}(g)$$

Hence, when the empirical risk $\widehat{\mathcal{R}}(g)$ is above the threshold ϵ, the modified estimator equals the empirical risk.

2) **When $\widehat{\mathcal{R}}(g) < \epsilon$, then**

$$|\widehat{\mathcal{R}}(g) - \epsilon| = \epsilon - \widehat{\mathcal{R}}(g)$$

$$\widetilde{\mathcal{R}}(g) = \epsilon - \widehat{\mathcal{R}}(g) + \epsilon = 2\epsilon - \widehat{\mathcal{R}}(g)$$

This effectively "reflects" the empirical risk $\widehat{\mathcal{R}}(g)$ about the level ϵ, penalizing overly optimistic (too low) risk estimates.

When the empirical risk $\widehat{\mathcal{R}}(g)$ is below the threshold ϵ, it increases the estimated risk, while if the empirical risk is above ϵ, it remains unchanged. This strategy helps achieve a more reliable and stable estimate of the true risk $R(g)$.

Question 4.3.b

The mean squared error (MSE) of any estimator E is given by

$$\mathrm{MSE}(E) = \mathbb{E}\Big[(E - R(g))^2\Big] = \big(\mathrm{Bias}(E)\big)^2 + \mathrm{Var}(E)$$

Since $\widehat{\mathcal{R}}(g)$ is unbiased, which means its average is exactly $\mathcal{R}(g)$, we can say that

$$\mathrm{MSE}\big(\widehat{\mathcal{R}}(g)\big) = \mathrm{Var}\big(\widehat{\mathcal{R}}(g)\big).$$

The modified estimator $\widetilde{\mathcal{R}}(g)$ may introduce a bias b but reduces the variance. Its MSE becomes

$$\mathrm{MSE}\big(\widetilde{\mathcal{R}}(g)\big) = b^2 + \mathrm{Var}\big(\widetilde{\mathcal{R}}(g)\big).$$

Its bias is given by

$$b = \mathbb{E}\big[\widetilde{\mathcal{R}}(g)\big] - R(g) = \mathbb{E}\Big[\big|\widehat{\mathcal{R}}(g) - \epsilon\big|\Big] + \epsilon - R(g)$$

If the reduction in variance is significant enough such that

$$b^2 + \mathrm{Var}\big(\widetilde{\mathcal{R}}(g)\big) < \mathrm{Var}\big(\widehat{\mathcal{R}}(g)\big)$$

- The term $\epsilon - R(g)$ measures how far our chosen constant ϵ is from the true risk.
- The term $\mathbb{E}\Big[\big|\widehat{\mathcal{R}}(g) - \epsilon\big|\Big]$ measures the average absolute deviation of the standard estimator from ϵ.

If ϵ is chosen very close to $R(g)$, then $\epsilon - R(g)$ is small. Additionally, if $\widehat{\mathcal{R}}(g)$ is symmetrically distributed around ϵ, the expected absolute deviation will be minimized, and thus the overall bias b will be small.

Question 4.3.c

The constant ϵ should be ideally chosen close to the true risk $R(g)$. When ϵ is well-tuned (i.e. $\epsilon \approx R(g)$), the bias b in $\widetilde{\mathcal{R}}(g)$ is minimized and the reflection $|\widehat{\mathcal{R}}(g) - \epsilon| + \epsilon$ prevents extreme underestimation, reducing variance.

Thus, a more accurate choice of ϵ lowers the overall MSE, making the modified estimator more effective.