CSCI 567, Spring 2023
Yan Liu

# Sample Quiz 1 - For Practice

Date: Feb 14, 2023
Duration: 90 minutes
Total scores: 100 points

| Problem | 1 | 2 | 3 | 4 | Total |
|---|---|---|---|---|---|
| Max | 40 | 20 | 20 | 20 | 100 |
| Points | | | | | |

## Instructions

**General:**

- The exam has a total of **8 pages** (including this cover). Each problem has several questions. Once you are permitted to open your exam (and not before), you should check and make sure that you are not missing any pages.
- Duration of the exam is **90 minutes**. Questions are not ordered by their difficulty. Budget your time on each question carefully. **Ask a proctor** if you have any question regarding the exam.
- Select **one and only one answer** for all multiple choice questions.
- Answers should be **concise** and written down **legibly**.
- You must answer on the page of each question. You can use the last blank page as scratch paper.
- This is a **closed-book/notes** exam. Consulting any resources is NOT permitted.
- Any kind of cheating will lead to **score 0** for the entire exam and be reported to SJACS.
- **(For DEN students only)** For all questions, write down your answers as you would normally do for the written assignments, and then submit a pdf/jpg/png for each of the 5 problems. Make sure that your writing is recognizable. Try to keep your solutions concise.

**Notes on notation:**

- Unless stated otherwise, scalars are denoted by small letter in normal font, vectors are denoted by small letters in bold font and matrices are denoted by capital letters in bold font.
- $\|.\|$ means L2-norm unless specified otherwise i.e. $\|.\| = \|.\|_2$

## Problem 1 Multiple Choice Questions (40 points)

Select **one and only one answer** for all multiple choice questions.

1. Which one of these is a sign of overfitting?

   (A) Low training error, low test error

   (B) Low training error, high test error

   (C) High training error, low test error

   (D) High training error, high test error

   Ans: B

2. Which of the following can help prevent overfitting?

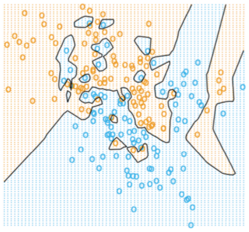
   (A) Using more complex models

   (B) Training until you get the smallest training error

   (C) Including a regularization coefficient on the loss function

   (D) All of the above

   Ans: C



3. Which of these classifiers could have generated this decision boundary?

   (A) Regularized Linear Regression

   (B) Perceptron

   (C) 1-nearest-neighbor

   (D) None of the above

   Ans: C

4. Suppose we are training a neural network with mini-batch SGD of batch size 50, and 50000 training samples. How many updates would there be while training during 5 epochs?

   (A) 50000

   (B) 1000

   (C) 5000

   (D) 250000

   Ans: C

5. Given a dataset which consists of a training set and a development set for hyperparameter (k) tuning. When we see that choosing a specific k results in very low training error but very high testing error, it is a good sign of underfitting.

(A) True

(B) False

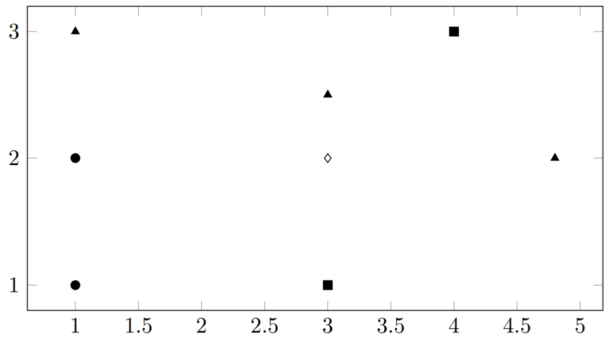
Ans: B

6. Because kNN is a very flexible non-parametric classifier, it can achieve near-perfect classification even for problems in which the true underlying data distributions overlap.

(A) True

(B) False

Ans: B



7. Consider the following two-dimensional dataset with N = 7 training points of three classes (triangle, square, and circle), and additionally one test point denoted by the diamond. Which of the following configuration of the K-nearest neighbor algorithm will **NOT** predict triangle for the test point?

(A) K = 1, L2 distance

(B) K = 3, L1 distance

(C) K = 3, L2 distance.

(D) K = 7, any distance.

Ans: C

8. Consider a training set $(x_1, y_1), \ldots, (x_N, y_N)$ and a probabilistic model $\mathbb{P}(y_n|\boldsymbol{x}_n; \boldsymbol{w})$ which specifies for each $n$ the probability of seeing outcome $y_n$ given feature $\boldsymbol{x}_n$ and parameter $\boldsymbol{w}$. Which of the following is the Maximum Likelihood Estimation (MLE) for $w$?

(A) $\arg\max \sum_{n=1}^{N} \mathbb{P}(y_n|\boldsymbol{x}_n; \boldsymbol{w})$

(B) $\arg\max \prod_{n=1}^{N} \mathbb{P}(\boldsymbol{x}_n; \boldsymbol{w})$

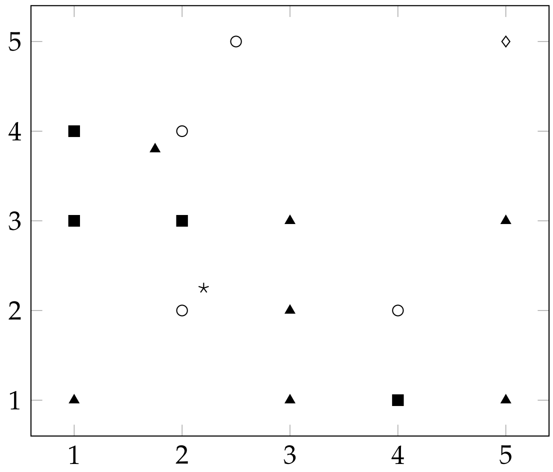
(C) $\arg\max \prod_{n=1}^{N} \ln \mathbb{P}(y_n|\boldsymbol{x}_n; \boldsymbol{w})$

(D) $\arg\max \sum_{n=1}^{N} \ln \mathbb{P}(y_n|\boldsymbol{x}_n; \boldsymbol{w})$

Ans: D

## Problem 2 Nearest Neighbor Classifier (20 points)

For the data given below, squares, triangles, and open circles are three different classes of data in the training set and the diamond (◇) and star (*) are test points. We denote the total number of training points as $N$ and consider K-nearest-neighbor (KNN) classifier with L2 distance.



(a) When $K = 1$, how many *circles*(o) will be misclassified (as a validation point) when one performs leave-one-out validation (that is, $N$-fold cross validation)? (A single number as the answer is enough.)

3

(b) What is the minimum value of $K$ for which the *star*(*) will be classified as a *triangle*? (A single number as the answer is enough.)

4

(c) What is the *diamond* classified as for $K = N$? Explain why.

Triangle

When $K = N$ the prediction is always the majority label of the training set. Triangle is the majority in this example.

## Problem 3 Linear Regression (20 points)

For a training set $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N) \in \mathbb{R}^D \times \mathbb{R}$, let $\boldsymbol{w}_*$ be the least square solution with no regularization (assume $\mathbf{X}^T\mathbf{X}$ is invertible where $\mathbf{X} \in \mathbb{R}^{N \times D}$ is the data matrix with each row corresponding to the feature of an example, as used in the class). Find the least square solution (with no regularization again) of the following new training set: $(\boldsymbol{x}_1, y_1 - \boldsymbol{w}_*^T\boldsymbol{x}_1), \ldots, (\boldsymbol{x}_N, y_N - \boldsymbol{w}_*^T\boldsymbol{x}_N)$.

Using the formula derived in the class, we know the new least square solution $\boldsymbol{w}'_*$ is

$$\boldsymbol{w}'_* = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}(\boldsymbol{y} - \mathbf{X}\boldsymbol{w}_*) = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\boldsymbol{y} - (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{X}((\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\boldsymbol{y}) = \mathbf{0}.$$

Rubrics:

- Write down the correct formula for $\boldsymbol{w}_*$.
- Use the correct formula to find $\boldsymbol{w}'_*$ or re-derive in a correct way (e.g. correct gradient).
- Get the correct answer finally $\boldsymbol{w}'_* = \mathbf{0}$.

## Problem 4 Perceptron (20 points)

Recall that a linear model for a multiclass classification problem with $C$ classes is parameterized by $C$ weight vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_C \in \mathbb{R}^D$. In the class we derive the multiclass logistic regression by minimizing the multiclass logistic loss. In this problem you need to derive the multiclass perceptron algorithm in a similar way. Specifically, the multiclass perceptron loss on a training set $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N) \in \mathbb{R}^D \times [C]$ is defined as

$$F(\boldsymbol{w}_1, \ldots, \boldsymbol{w}_C) = \frac{1}{N} \sum_{n=1}^{N} \max\left\{0, \max_{k \neq y_n} \boldsymbol{w}_k^{\mathrm{T}} \boldsymbol{x}_n - \boldsymbol{w}_{y_n}^{\mathrm{T}} \boldsymbol{x}_n\right\}.$$

Similarly to the binary case, multiclass perceptron is simply applying SGD with learning rate 1 to minimize the multiclass perceptron loss. Based on the above information, write down the multiclass perceptron algorithm below. (For simplicity, you do not need to worry about the non-differential points of $F$. In other words, the term $\max_{k \neq y_n} \boldsymbol{w}_k^{\mathrm{T}} \boldsymbol{x}_n - \boldsymbol{w}_{y_n}^{\mathrm{T}} \boldsymbol{x}_n$ is never 0.)

Solutions:

**Algorithm 1** Multiclass Perceptron

```
Input: A training set $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)$
Initialize: $\boldsymbol{w}_1 = \cdots = \boldsymbol{w}_C = 0$ (or randomly)

while not converged do
    randomly pick an example $(\boldsymbol{x}_n, y_n)$, make prediction $\hat{y} = \arg\max_{k \in [C]} \boldsymbol{w}_k^{\mathrm{T}} \boldsymbol{x}_n$
    if $\hat{y} \neq y_n$ then
        $\boldsymbol{w}_{\hat{y}} \leftarrow \boldsymbol{w}_{\hat{y}} - \boldsymbol{x}_n$

        $\boldsymbol{w}_{y_n} \leftarrow \boldsymbol{w}_{y_n} + \boldsymbol{x}_n$
```

Deriving the gradients correctly but not putting everything into an algorithm framework still gets full points.