CPSC 340 2021S Final Exam (Real Thing)

IMPORTANT: Download the .tex source file here!!!!

https://www.students.cs.ubc.ca/~cs-340/namhee/final_exam.tex

Instructions

As with the assignments, please indicate your name and 8-digit UBC student ID number in your solution.

- You have 24 hours to complete this exam. You can submit anytime, however many times during the exam window.
- You may submit a handwritten solution instead of a typeset document. If you do so, please clearly mark your answers with corresponding question numbers.
- You do not need to show your work unless specifically instructed, but we will not award partial credits for incorrect solutions if you don't show your work.
- The exam is open book, meaning you are allowed to consult course materials, the internet, etc.
- You may NOT communicate with anyone else (other than the instructor) in any way during the exam. This includes posting anything on the internet (e.g. **Discord**, a message board, chat, tutoring service, etc.) during the exam. UBC has a strict policy on academic misconduct, along with disciplinary measures, which I can attest are not fun for anyone involved.
- You may NOT copy/paste solutions from anywhere. If you consulted any online resources and wish to cite this, feel free to drop a link in your exam submission to be safe.
- Announcements or clarifications will be made on a Piazza thread. Look out for the official final exam post.
- If you have a question, make a private post on Piazza.
- Do not post or communicate about the exam until at least 1 hour after the scheduled end time.

The submission format for this final exam is identical to the submission format for the homework assignments, except two things: (1) you cannot work with a partner, and (2) you can submit handwritten solutions.

Please follow the official submission instructions. In short, submit to Gradescope and match the questions to pages when prompted by Gradescope. We reserve the rights to deduct 5 points off of your submission if you do not follow the submission instructions.

1 Very-Short Answer Questions

Rubric: {points:22}

Answer the following questions in 1-2 *brief* sentences. Please avoid writing long answers.

1. Recall the identical and independent distribution (IID) assumption. Describe why non-IID data can be problematic for maximum likelihood estimation.

   Answer: The total likelihood $p(y \mid x_i, w)$ is expressed as a product across each example's individual likelihood $p(y_i \mid x_i, w)$ thanks to the IID assumption. Without the IID assumption, we cannot do this.

2. Recall cross-validation. Describe one key difference between using a validation set and using cross-validation.

   Answer: Cross-validation uses many validation sets based on different folds.

3. Recall feature standardization and decision trees. For a dataset with continuous features and discrete labels, describe the effect of feature standardization on the classification performance of a decision stump.

   Answer: No change. Decision stumps will have the exact same behaviour for linearly-scaled features.

4. Recall L2-regularization and logistic regression. Suppose an example x_i is classified as $\hat{y}_i = +1$ without L2-regularization. Describe the effect of λ, the strength of L2-regularization, on $p(y_i = +1 \mid x_i, w)$, the sigmoid likelihood that the example x_i is positive.

   Answer: As λ increases, the likelihood decreases, because the resulting line or hyperplane corresponding to w will have a less steep slope, resulting in a decreased magnitude for $w^T x_i$.

5. Recall the loss function for a support vector machine (SVM) and a logistic regression model. Describe one key difference between SVM's loss function and logistic regression's loss function.

   Answer: SVM's loss function is hinge loss, which is still non-smooth. Logistic regression uses logistic loss, which is smooth.

6. Recall using local and global features. Describe one advantage of using local features.

   Answer: We can produce personalized or localized predictions by adding or subtracting from the global features-based predictions.

7. Recall stochastic gradient descent. Describe one advantage of using a constant step size $\alpha^t = c$ for stochastic gradient steps.

   Answer: Constant step size will get to the ball of confusion very fast, and it's a reasonable thing to do if we don't want absolutely best precision.

8. Recall using n-grams to represent text data features. Describe one disadvantage of increasing n, the number of consecutive words captured into the columns of bag-of-words.

   Answer: The coupon collector problem becomes worse. We will need even more examples in order to see each possible configuration of bag-of-words at least once.

9. Recall the fundamental trade-off and matrix factorization for collaborative filtering: $X = ZW$ where X is centered and has missing values. Describe the effect of increasing k, the number of components on (1) reconstruction error on available values and (2) how well that reconstruction error estimates "prediction error" for missing values.

   Answer: As k increases, reconstruction error decreases and prediction error increases. When $k = 0$, the missing values are replaced by the entry-wise mean of X. When $k = d$, the missing values are predicted by a dot product between large vectors, which is subject to higher prediction error.

10. Recall manifolds. Suppose I have a labeled dataset, whose examples are distributed in a low-dimensional non-linear manifold within the d-dimensional feature space. Describe one way to perform classification on this data, taking the manifold structure into account.

    Answer: Use neural networks, k-nearest neighbours, or use ISOMAP to map the datapoints into a Euclidean space and then learn a linear model.

11. Recall artificial neural networks. Describe one advantage of using a "deeper" encoder of low-dimensional hidden features as opposed to using a single-layer encoder of high-dimensional features.

    Answer: Deeper neural networks can establish a hierarchy among learned features while using fewer parameters for matrix multiplication.

2 Kernelized Decision Trees

Rubric: {points:15}

Suppose I have the following training data with continuous features and categorical labels:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & -1 \\ -1 & 1 \end{bmatrix} \qquad y = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

Suppose I have a particular kernel function K where $K(x_i, x_j) = z_i^T z_j$ and z_i is the result of using a change-of-basis with these features (including the coefficients):

$$z_i = \begin{bmatrix} 1 & \sqrt{2}x_{i1} & \sqrt{2}x_{i2} & x_{i1}^2 & \sqrt{2}x_{i1}x_{i2} & x_{i2}^2 \end{bmatrix}$$

1. Compute the resulting kernel matrix K, where an entry $k_{ij} = K(x_i, x_j)$.

    Answer: Using the kernel trick, we get $K = (1 + XX^T)^2$.

$$XX^T = \begin{bmatrix} 1 & -1 & 1 \\ -1 & 2 & 0 \\ 1 & 0 & 2 \end{bmatrix}$$

$$1 + XX^T = \begin{bmatrix} 2 & 0 & 2 \\ 0 & 3 & 1 \\ 2 & 1 & 3 \end{bmatrix}$$

$$\left(1 + XX^T\right)^2 = \begin{bmatrix} 4 & 0 & 4 \\ 0 & 9 & 1 \\ 4 & 1 & 9 \end{bmatrix}$$

2. Using the greater-than-or-equal-to ($\geq$) splitting rule on the third feature ($j = 3$) of the kernel matrix K, create a decision stump that maximizes classification accuracy. What is the splitting rule? What are the resulting y_{yes} and y_{no}?

    Answer: Splitting rule is $k_{i3} \geq 4$. y_{yes} is $\begin{bmatrix} 0 & 0 \end{bmatrix}^T$, and y_{no} is $\begin{bmatrix} 1 \end{bmatrix}^T$

3. Remove the incorrect option in each parenthesis so that the resulting sentence is true:

    Answer: This stump is a (parametric / **non-parametric**) model because the (**memory** / time) complexity of the model (**does** / doesn't) grow with the number of (**examples** / features).

3 k-means Clustering and Vector Quantization

Rubric: {points: 10}

Suppose I have the following unsupervised training data

$$X = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 0 & -1 \\ 2 & 0 \\ -2 & 0 \end{bmatrix}$$

To perform k-means clustering with $k = 3$, I initialize my matrix W as follows:

$$W = \begin{bmatrix} 0 & 0 \\ 3 & 0 \\ -3 & 0 \end{bmatrix}$$

1. Write down the converged W from running k-means clustering with the initial W on the matrix X.

   Answer:

   $$W = \begin{bmatrix} 0 & 0 \\ 2 & 0 \\ -2 & 0 \end{bmatrix}$$

   The first mean stays the same, the second and the third means move to their closest examples.

2. Suppose I have a test example $\tilde{x}_i = \begin{bmatrix} 0 & 2 \end{bmatrix}^T$. Perform vector quantization on $\tilde{x}_i$ using the converged W of k-means clustering to obtain a $d \times 1$ vector $\hat{x}_i$, the compressed version of $\tilde{x}_i$. What is the resulting $\hat{x}_i$?

   Answer: $\tilde{x}_i$ is closest to the first mean.

   $$\hat{x}_i = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

   Also accepted:

   $$\tilde{z}_i = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

4 Designing Objective Functions

Rubric: {points: 20}

Using the vector and matrix notation covered in the course, write down the objective function f for the following items, minimizing which will achieve each described model/behaviour. For example,

- An ordinary least squares linear regression

  Answer:

  $$f(w) = \frac{1}{2}||Xw - y||^2$$

1. A least squares linear regression model with a sparse solution, whose sparsity is controlled by a positive scalar λ.

   Answer:
   $$f(w) = \frac{1}{2}||Xw - y||^2 + \lambda||w||_1$$
   $\frac{1}{2}$ is optional. L0-penalty is also acceptable.

2. A linear binary classifier with a sparse solution, whose (1) loss function is an upper convex envelope of the 0-1 loss and (2) sparsity is controlled by a positive scalar λ. Assume $y_i \in \{-1, +1\}$.

   Answer:
   $$f(w) = \sum_{i=1}^{n} \max\{0, 1 - y_i w^T x_i\} + \lambda||w||_1$$
   L0-penalty is also acceptable.

3. A brittle PCA: a matrix factorization model performing $X \approx ZW$, where Z is an $n \times k$ matrix and W is a $k \times d$ matrix. This objective function is continuous, differentiable, and extremely sensitive to any outlier entry in the matrix X.

   Answer:
   $$f(Z, W) = \log\left(\sum_{i=1}^{n}\sum_{j=1}^{d} \exp\left(\langle w^j, z_i\rangle - x_{ij}\right) + \exp\left(x_{ij} - \langle w^j, z_i\rangle\right)\right)$$
   $\exp\left(\langle w^j, z_i\rangle - x_{ij}\right)^2$, $\exp\left(|\langle w^j, z_i\rangle - x_{ij}|\right)$ and $\exp\left(\langle w^j, z_i\rangle - x_{ij}\right)$ are also accepted, since we didn't make it explicit how to handle negative residuals. I did something like $\exp\left(|\langle w^j, z_i\rangle - x_{ij}|\right)$ in the practice midterm and now I realize it's not differentiable at 0 :(

4. A multi-dimensional scaling model that (1) visualizes the Euclidean distance in the d-dimensional feature space of X as the Manhattan distance in the 2-dimensional learned feature space with Z and (2) minimizes the squared difference between the two distances.

   Answer:
   $$f(Z) = \sum_{j>i=1}^{n} \left(||x_i - x_j||_2 - ||z_i - z_j||_1\right)^2$$
   $\sum_{j>i=1}^{n}$ is just a short-hand notation for "sum over all pairs". It's fine if the solution has squared L2-norm for Euclidean distance.

Hints:

- You can write your solutions with the $\sum$ notation instead of expressing everything as matrix-vector products.
- Remember that matrices don't have the same norms as vectors.

5 MAP Estimation

Rubric: {points:10}

Consider using the Laplace likelihood with a mean $w^T x_i$ and example-specific scale $b_i > 0$, and we use a non-zero mean Gaussian prior with a variance σ^2:

$$p(y_i \mid x_i, w) = \frac{1}{2b_i}\exp\left(-\frac{|w^T x_i - y_i|}{b_i}\right), \qquad p(w_j) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(w_j - \mu_j)^2}{2\sigma^2}\right)$$

1. Find an objective function $f(w)$ whose minimizer corresponds to the maximum a posteriori (MAP) estimate of the resulting posterior distribution.

   Answer: Taking the negative log likelihood: with IID assumption, I get

$$
\begin{aligned}
p(y \mid X, w) &= \prod_{i=1}^{n} \frac{1}{2b_i} \exp\left(-\frac{|w^T x_i - y_i|}{b_i}\right) \\
-\log p(y \mid X, w) &= -\sum_{i=1}^{n} \left[\log\left(\frac{1}{2b_i}\right) - \frac{|w^T x_i - y_i|}{b_i}\right] \\
&\approx \sum_{i=1}^{n} \frac{|w^T x_i - y_i|}{b_i} \\
&= ||B^{-1}(Xw - y)||_1.
\end{aligned}
$$

   Taking the negative log prior: again, with IID assumption, I get

$$
\begin{aligned}
p(w) &= \prod_{j=1}^{d} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(w_j - \mu_j)^2}{2\sigma^2}\right) \\
-\log p(w) &= \left[-\sum_{j=1}^{d} \log\left(\frac{1}{\sqrt{2\pi}\sigma}\right) - \frac{(w_j - \mu_j)^2}{2\sigma^2}\right] \\
&\approx \sum_{j=1}^{d} \frac{(w_j - \mu_j)^2}{2\sigma^2} = \frac{1}{2\sigma^2} \sum_{j=1}^{d} (w_j - \mu_j)^2 \\
&= \frac{1}{2\sigma^2} ||w - m||^2
\end{aligned}
$$

   Putting them together:

$$
f(w) = ||B^{-1}(Xw - y)||_1 + \frac{1}{2\sigma^2} ||w - m||^2
$$

2. Describe the effect of increasing σ^2, the shared variance of the parameters, on the two parts of the fundamental trade-off.

   Answer: As σ^2 increases, the strength of regularization decreases, and therefore training error decreases and approximation error increases.

Hint: you may use:

- B, an $n \times n$ diagonal matrix whose ith diagonal value is b_i.
- m, a $d \times 1$ vector whose jth entry is μ_j.

6 Projected Gradient

Rubric: {points:10}

Suppose we have the following labeled data with continuous features and labels:

$$
X = \begin{bmatrix} 3 & 1 \\ 1 & 1 \end{bmatrix} \qquad y = \begin{bmatrix} -1 \\ 2 \end{bmatrix}.
$$

Suppose we want to minimize the least squares objective while constraining my solution w to be non-negative:

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}\left(w^T x_i - y_i\right)^2.$$

We are given a constant step size $\alpha^t = 1$ and the initial guess $w^0 = \begin{bmatrix}0 & 0\end{bmatrix}^T$ for gradient descent.

Compute w^1 by performing a projected gradient step.

Answer: The least squares objective's gradient is

$$\nabla f(w) = X^T X w - X^T y = X^T(Xw - y).$$

The first half-step is regular gradient descent step.

$$\begin{aligned}
\nabla f(w^0) &= X^T(Xw^0 - y) \\
&= -X^T y \\
&= -\begin{bmatrix}3 & 1\\ 1 & 1\end{bmatrix}\begin{bmatrix}-1\\ 2\end{bmatrix} = \begin{bmatrix}1\\ -1\end{bmatrix} \\
w^{\frac{1}{2}} &= w^0 - \alpha^t \nabla f(w^0) \\
&= \begin{bmatrix}0\\ 0\end{bmatrix} - 1 \cdot \begin{bmatrix}1\\ -1\end{bmatrix} = \begin{bmatrix}-1\\ 1\end{bmatrix}
\end{aligned}$$

The second half step takes element-wise max operation to mask non-negatives as zeros. Then:

$$w^1 = \begin{bmatrix}0\\ 1\end{bmatrix}$$

7 Softmax Classifier

Rubric: {points:15}

Recall multi-class logistic regression. Suppose I fit a softmax classifier for a particular labeled dataset with continuous features and discrete labels, where $y_i \in \{1, 2, 3\}$:

$$X = \begin{bmatrix}0 & 1\\ 1 & 0\\ 1 & 0\\ 1 & 2\\ -1 & -1\end{bmatrix} \qquad y = \begin{bmatrix}1\\ 2\\ 2\\ 2\\ 3\end{bmatrix}$$

and obtained a solution

$$W = \begin{bmatrix}0 & 1\\ 1 & 0\\ -1 & -1\end{bmatrix}.$$

This W is obtained from running an optimizer on the softmax loss function:

$$f(W) = \sum_{i=1}^{n} \left[-w_{y_i}^T x_i + \log\left(\sum_{c=1}^{k} \exp(w_c^T x_i) \right) \right].$$

Answer the following questions:

1. What is the baseline training classification accuracy for this classification problem?

   Answer: If we return the mode and predict $\hat{y}_i = 2$ every time, we get the accuracy of 3/5, so 60%.

2. What is the training classification accuracy for W?

   Answer:

   $$XW^T = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 2 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} 0 & 1 & -1 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ 0 & 1 & -1 \\ 2 & 1 & -3 \\ -1 & -1 & 2 \end{bmatrix}$$

   Taking the argmax for each row, we get

   $$\hat{y} = \begin{bmatrix} 1 \\ 2 \\ 2 \\ 1 \\ 3 \end{bmatrix}$$

   The classification accuracy for W is 4/5, so 80%.

3. Suppose I have a test example $\tilde{x}_i = \begin{bmatrix} -1 & 1 \end{bmatrix}^T$. What is the probability that this test example's label is $\tilde{y}_i = 1$?

   Answer:

   $$W\tilde{x}_i = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$$

   Using softmax, we compute the conditional probability:

   $$p(y_i = +1 | x_i, W) = \frac{\exp(1)}{\exp(1) + \exp(-1) + \exp(0)}$$

Hint: you can leave the exponential functions unevaluated in your solution.

8 Convex Functions

Rubric: {points:15}

Consider a convex approximation to the L1-loss regression objective, which has the following form:

$$f(w) = \sum_{i=1}^{n} \sqrt{(w^T x_i - y_i)^2 + \epsilon}$$

where w is a $d \times 1$ vector, and x_i and y_i are our usual features and label for example i, respectively.

1. Show that $(w^T x_i - y_i)^2 + \epsilon$ is (1) convex and (2) greater than 0 (assume $\epsilon > 0$) for any w, x_i and y_i.

   Answer: $w^T x_i - y_i$ is a linear mapping, and squaring a linear mapping preserves convexity, hence $(w^T x_i - y_i)^2$ is convex. Adding a constant preserves convexity, so $(w^T x_i - y_i)^2 + \epsilon$ is convex. Squaring a variable is always non-negative, and $\epsilon > 0$, hence $(w^T x_i - y_i)^2 + \epsilon > 0$.

2. Show that $g(r) = \sqrt{r^2 + \epsilon}$ is convex (assume $\epsilon > 0$) for any r.

   Answer: Taking the first derivative:

$$\begin{aligned}\frac{\partial g}{\partial r} &= \frac{\partial}{\partial r}\left(r^2 + \epsilon\right)^{\frac{1}{2}} \\ &= \frac{1}{2}\left(r^2 + \epsilon\right)^{-\frac{1}{2}} \cdot 2r \\ &= r\left(r^2 + \epsilon\right)^{-\frac{1}{2}}\end{aligned}$$

   Taking the second derivative:

$$\begin{aligned}\frac{\partial^2 g}{\partial r^2} &= \frac{\partial}{\partial r} r\left(r^2 + \epsilon\right)^{-\frac{1}{2}} \\ &= r \cdot -\frac{1}{2}(r^2 + \epsilon)^{-\frac{3}{2}} \cdot 2r + \left(r^2 + \epsilon\right)^{-\frac{1}{2}} \\ &= -\frac{r^2}{(r^2 + \epsilon)^{\frac{3}{2}}} + \frac{1}{(r^2 + \epsilon)^{\frac{1}{2}}}.\end{aligned}$$

   Multiplying the top and bottom by $(r^2 + \epsilon)$ to the right term, I get

$$\begin{aligned}\frac{\partial^2 g}{\partial r^2} &= \frac{-r^2 + r^2 + \epsilon}{(r^2 + \epsilon)^{\frac{3}{2}}} \\ &= \frac{\epsilon}{(r^2 + \epsilon)^{\frac{3}{2}}}\end{aligned}$$

   Since ϵ is positive, the second derivative is positive everywhere. Hence $g(r)$ is convex.

3. Compute the gradient of f. You may write it as a partial derivative with respect to some w_j, although it is possible to write it in the matrix-vector notation.

   Answer: Taking the partial derivative with respect to some w_j:

$$\begin{aligned}\frac{\partial f}{\partial w_j} &= \frac{1}{2}\sum_{i=1}^{n}\left((w^T x_i - y_i)^2 + \epsilon\right)^{-\frac{1}{2}} \cdot 2(w^T x_i - y_i)x_{i1} \\ &= \sum_{i=1}^{n}\frac{(w^T x_i - y_i)x_{ij}}{\sqrt{(w^T x_i - y_i)^2 + \epsilon}}\end{aligned}$$

   It is possible to express this in the matrix-vector notation. We should define an $n \times 1$ vector r, such that

$$r_i = \frac{(w^T x_i - y_i)}{\sqrt{(w^T x_i - y_i)^2 + \epsilon}}.$$

   Then

$$\nabla f(w) = X^T r.$$

9 Runtime Complexity Analysis

Rubric: {points:10}

Answer the following questions using the big-O notation.

1. Consider the following outlier detection method based on PCA computed via SVD (hence W is orthogonal). We are given a trained PCA model which has k components. For a test example $\tilde{x}_i$ (a $d \times 1$ vector) we compute $\tilde{z}_i$ and then reconstruct $\hat{x}_i$. If the L2-distance between $\tilde{x}_i$ and $\hat{x}_i$ is above some threshold ϵ (i.e. $||\tilde{x}_i - \hat{x}_i|| > \epsilon$), then we declare that $\tilde{x}_i$ is an outlier. What is the running time for determining whether **one test example** is an outlier? Your answer may or may not depend on n, d, k, and/or ϵ.

   Answer: $O(kd)$. First, we center $\tilde{x}_i$ in $O(d)$ time. Since we used SVD and thus W is orthogonal, $\tilde{z}_i = W\tilde{x}_i$. This involves a matrix-vector multiplication between a $k \times d$ matrix and a $d \times 1$ vector, resulting in a $O(kd)$ cost. Reconstructing $\hat{x}_i = W^T\tilde{z}_i$ similarly results in a $O(kd)$ cost. Computing distance between $\tilde{x}_i$ and $\hat{x}_i$ is an $O(d)$ operation, and comparison is constant time.

2. Consider performing a change-of-basis with the Gaussian RBF function and then training an RBF regression model on a training dataset that has n examples and d features. What is the cost of applying the normal equations to solve the RBF regression? Your answer may or may not depend on n and/or d.

   Answer: $O(n^2d + n^3)$. Forming the $n \times n$ matrix Z takes $O(n^2d)$ time. With n features, the normal equations takes $O(n^3)$ to solve.

10 Memory Complexity Analysis

Rubric: {points:10}

Answer the following questions using the big-O notation.

1. Given an $n \times d$ matrix X as our unsupervised training data, what is the memory complexity required for storing the parameters of a multi-dimensional scaling model with k-dimensional learned features? That is, if we wanted to encode new examples in a $t \times d$ matrix $\tilde{X}$ with a "learned" MDS model, how much memory are we required to ship? Your answer may or may not depend on n, t, d, and/or k.

   Answer: $O(nd)$. We need to ship the entire dataset. $O(nd + nk)$ is also acceptable if any sort of "warm-start" is mentioned.

2. Consider a neural network regression model with the following properties: (1) the encoder has m layers, (2) each layer outputs k-dimensional features and uses a sigmoid activation, and (3) the predictor generates the scalar prediction $\hat{y}_i$ for a given $d \times 1$ example $\tilde{x}_i$. What is the memory complexity required for storing the parameters of this model? You can ignore biases. Your answer may or may not depend on n, t, d, and/or k.

   Answer: $O(dk + mk^2)$. The first layer of the encoder is a $d \times k$ matrix, and from there we have $m - 1$ layers that are $k \times k$ matrices. Activation function does not store any parameter. The predictor's weights are stored into $O(k)$ parameters.

11 Objective Function: "Max"-Rule Multi-Class Hinge Loss

Rubric: {points: 20}

Recall the constraints we introduce for multi-class linear classifiers. Plainly, for an example whose label is y_i, we want the linear score for y_i to be larger than the linear scores for other classes.

More formally, we want:

$$w_{y_i}^T x_i > w_{c'}^T x_i, \quad y_i \neq c' \in \{1, \cdots, k\}, \tag{1}$$

where k is the number of classes for the labels.

Recall that counting the number of times this constraint is violated is how we derive the multi-class hinge loss, which yields a multi-class support vector machine. Plainly, we count the number of times Constraint (1) is violated. More formally, for each example i, whose label is y_i, we incur this penalty:

$$f_i(W) = \sum_{c' \neq y_i} I(w_{y_i}^T x_i \leq w_{c'}^T x_i). \tag{2}$$

Let $I(\cdot)$ be the indicator function which evaluates to 1 if the clause inside evaluates to true, and 0 otherwise. Our objective function is:

$$f(W) = \sum_{i=1}^{n} \sum_{c' \neq y_i} I(w_{y_i}^T x_i \leq w_{c'}^T x_i). \tag{3}$$

In the practice exam, we showed how a convex approximation for Equation (3) yields the "sum"-rule multi-class hinge loss.

Your task is to derive the "max"-rule multi-class hinge loss following the same steps.

Now, the way we count constraint violations is this: plainly, we penalize the model proportional to the "largest" violation of the constraint, instead of using 0-1 loss. More formally, for each example i, whose label is y_i, we incur this penalty:

$$f_i(W) = \max_{c' \neq y_i} \left[I(w_{y_i}^T x_i \leq w_{c'}^T x_i) \cdot (w_{c'}^T x_i - w_{y_i}^T x_i) \right]. \tag{4}$$

Using Equation (4) to count constraint violation, we will derive the "max"-rule multi-class hinge loss.

Answer the following questions:

1. Write down the objective function f that sums the constraint violations in Equation (4) over all examples in the training data.

   Answer: Our objective function is:

   $$f(W) = \sum_{i=1}^{n} \max_{c' \neq y_i} \left[I(w_{y_i}^T x_i \leq w_{c'}^T x_i) \cdot (w_{c'}^T x_i - w_{y_i}^T x_i) \right]. \tag{5}$$

2. Rearrange the expression $I(w_{y_i}^T x_i \leq w_{c'}^T x_i) \cdot (w_{c'}^T x_i - w_{y_i}^T x_i)$ as a max operation. Write down the objective function f incorporating this.

   Answer: First, we restructure what's inside the indicator.

   $$I(w_{y_i}^T x_i \leq w_{c'}^T x_i) = I(0 \leq -w_{y_i}^T x_i + w_{c'}^T x_i). \tag{6}$$

Then the expression becomes:

$$I(w_{y_i}^T x_i \leq w_{c'}^T x_i) \cdot (w_{c'}^T x_i - w_{y_i}^T x_i) = I(0 \leq -w_{y_i}^T x_i + w_{c'}^T x_i) \cdot (w_{c'}^T x_i - w_{y_i}^T x_i) \tag{7}$$

$$= I(0 \leq -w_{y_i}^T x_i + w_{c'}^T x_i) \cdot (-w_{y_i}^T x_i + w_{c'}^T x_i). \tag{8}$$

Note that I rearranged $(w_{c'}^T x_i - w_{y_i}^T x_i)$ to make it easier to recognize. Now, when $0 \leq -w_{y_i}^T x_i + w_{c'}^T x_i$, the penalty is equal to $-w_{y_i}^T x_i + w_{c'}^T x_i$. When $0 > -w_{y_i}^T x_i + w_{c'}^T x_i$, the penalty is 0. This behaviour is exactly the same thing as $\max\{0, -w_{y_i}^T x_i + w_{c'}^T x_i\}$. Therefore,

$$I(w_{y_i}^T x_i \leq w_{c'}^T x_i) \cdot (w_{c'}^T x_i - w_{y_i}^T x_i) = \max\{0, -w_{y_i}^T x_i + w_{c'}^T x_i\} \tag{9}$$

Then the objective function is

$$f(W) = \sum_{i=1}^{n} \max_{c' \neq y_i} \left[\max\{0, -w_{y_i}^T x_i + w_{c'}^T x_i\}\right]. \tag{10}$$

3. Use the "margin-of-1" trick to address the degeneracy when $W = 0$. Write down the objective function f incorporating this.

   Answer:

$$f(W) = \sum_{i=1}^{n} \max_{c' \neq y_i} \left[\max\{0, 1 - w_{y_i}^T x_i + w_{c'}^T x_i\}\right]. \tag{11}$$

4. Show that $f(W) = n$ when $W = 0$ with the "max"-rule multi-class hinge loss f (you can break ties arbitrarily).

   Answer:

$$f(0) = \sum_{i=1}^{n} \max_{c' \neq y_i} \left[\max\{0, 1\}\right] \tag{12}$$

$$= \sum_{i=1}^{n} 1 \tag{13}$$

$$= n. \tag{14}$$