USC CSCI 467
Intro to Machine Learning

Midterm Exam

October 10, 2023, 2:00-3:20pm

Fall 2023
Instructor: Robin Jia

Name: ____________________

USC e-mail: ____________________@usc.edu

Answer the questions in the spaces provided. **If you write solutions on the back of the pages, indicate this on the front of the pages so we know to look there, but please try to avoid this if possible.** You may use the backs of pages for scratch work. This exam has 5 questions, for a total of 100 points.

This content is protected and may not be shared, uploaded, or distributed.

Question 1: Weighted Linear Regression (28 points)

Consider a modification of the standard linear regression setup where each datapoint is associated with an importance weight. Formally, we have a training dataset consisting of n examples $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\}$, where each datapoint is associated with an importance weight $r_i > 0$. The weighted residual sum of squares objective is defined as

$$\text{WRSS}(w) = \sum_{i=1}^{n} r_i \left(w^\top x^{(i)} - y^{(i)}\right)^2.$$

(a) (5 points) Derive the expression for $\nabla \, \text{WRSS}(w)$.

Solution:

$$\nabla_w \, \text{WRSS}(w) = 2\sum_{i=1}^{n} r_i (w^\top x^{(i)} - y^{(i)}) \cdot x^{(i)}.$$

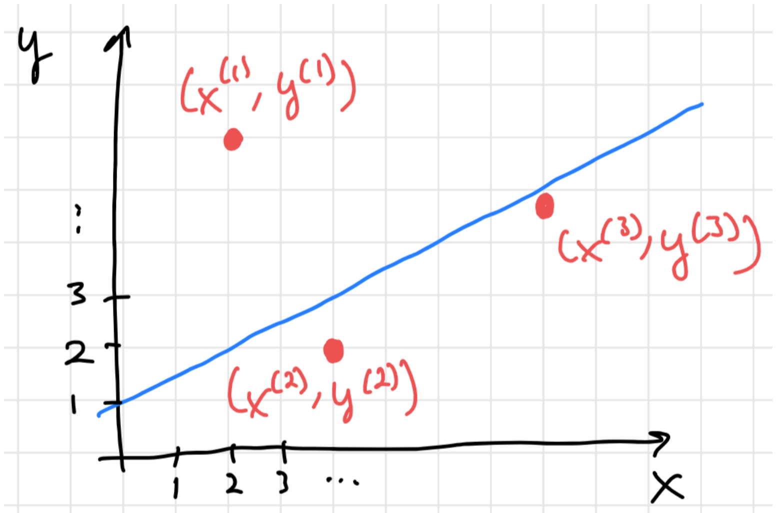
(b) (6 points) Let $\boldsymbol{X}$ be the $n \times d$ matrix whose i-th row is $x^{(i)\top}$, $\boldsymbol{y}$ be the n-dimensional column vector whose i-th entry is $y^{(i)}$, and $\boldsymbol{R}$ be the diagonal matrix where $\boldsymbol{R}_{ii} = r_i$ for all i and 0 for all other entries. Show that the WRSS objective can be written as follows in matrix form:

$$\text{WRSS}(\boldsymbol{w}) = (\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y})^{\mathrm{T}} \boldsymbol{R} (\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}).$$

Solution: Since the i-th row of $\boldsymbol{X}$ is $\boldsymbol{x}_i^{\mathrm{T}}$, $(\boldsymbol{X}\boldsymbol{w})_i = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i$.
As $\boldsymbol{R}$ is a diagonal matrix with $\boldsymbol{R}_{ii} = r_i$, $(\boldsymbol{R}(\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}))_i = r_i(\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y})_i$.
Using these, we get:

$$\begin{aligned}
\text{WRSS}(\boldsymbol{w}) &= \sum_{i=1}^{n} r_i \left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i - y_i\right)^2 \\
&= \sum_{i=1}^{n} \left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i - y_i\right) r_i \left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i - y_i\right) \\
&= \sum_{i=1}^{n} (\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y})_i (\boldsymbol{R}(\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y}))_i \\
&= (\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y})^{\mathrm{T}} \boldsymbol{R} (\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y})
\end{aligned}$$

(c) The diagram below shows a training dataset with three examples $(x^{(1)}, y^{(1)})$, $(x^{(2)}, y^{(2)})$, and $(x^{(3)}, y^{(3)})$.



Here the inputs x are all one-dimensional. Soumya chooses some importance weights r_1, r_2, and r_3 and then trains a model to minimize the WRSS objective. The model's predictions are illustrated by the blue line.

i. (2 points) What are the values of w and b that Soumya learned?

Solution: The blue line is given by the equation $y = wx + b$, so we can see from the graph that $w = \frac{1}{2}$ and $b = 1$.

ii. (4 points) Out of the values for r_1, r_2, and r_3 that Soumya chose, which one was the smallest? Explain your reasoning.

Solution: r_1 must be the smallest because the model has very large error on $(x^{(1)}, y^{(1)})$ compared with the other examples. This means that example 1 is the least important example, which corresponds to the smallest r_1.

(d) (3 points) Explain in 1-2 sentences how allowing $r_i < 0$ for some i could lead to undesirable behavior.

Solution: If $r_i < 0$, then the loss decreases as the error on the i-th example increases. So, the model is incentivized to make as large of error on that example as possible, which results in bad predictions.

(e) Let $w^\star$ be any value of w that achieves the lowest possible value of the standard linear regression loss discussed in class. Let $\tilde{w}$ be any value of w that achieves the lowest possible WRSS loss when $r_i = 10$ for all i.

i. (5 points) Assume that the matrix $\boldsymbol{X}^\top \boldsymbol{X}$ is invertible. Are $w^\star$ and $\tilde{w}$ guaranteed to be the same? Explain your reasoning.

Solution: Yes, $w^\star$ and $\tilde{w}$ are guaranteed to be the same. First, we recognize that when $X^\top X$ is invertible, there is a unique minimizer of the original linear regression loss function (which is given by the normal equations), so $w^\star$ must be this value. Second, we recognize that when $r_i = 10$ for all i, minimizing WRSS is equivalent to minimizing the original linear regression loss, since it's just the original loss multiplied by a positive constant. Thus, this also has the same unique solution, which must be equal to $\tilde{w}$. So, it must be that $w^\star = \tilde{w}$.

ii. (3 points) Now assume that $X^\top X$ is not invertible. Does this change your answer? Explain your reasoning.

Solution: This does change the answer—now $w^\star$ and $\tilde{w}$ are not guaranteed to be the same. If $X^\top X$ is not invertible, then there is no longer a unique solution to the linear regression minimization problem. So, it is possible that $w^\star$ and $\tilde{w}$ both achieve the minimum value, but they are not the same vector.

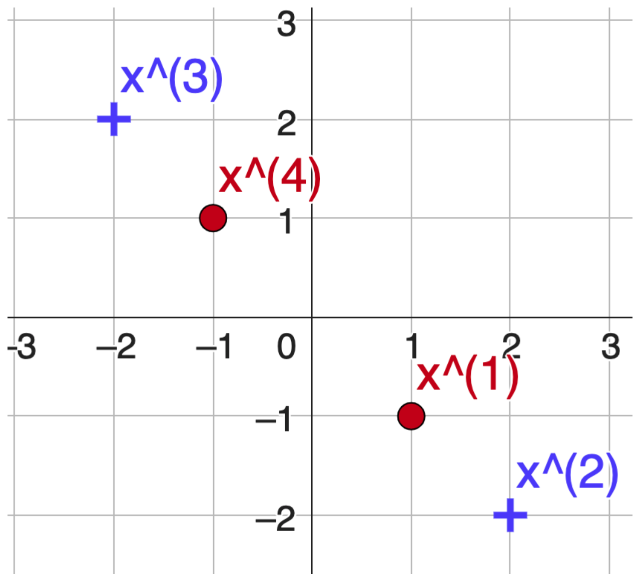
Question 2: Logistic Regression and Kernels (23 points)

In class, we discussed performing classification tasks using kernelized logistic regression. One popular choice of kernel function is the polynomial kernel, defined as

$$k(x, z) = (x^\top z + p)^q$$

where $p, q \in \mathbb{R}$ are hyperparameters.

In this problem, we will explore using this kernel for different values of p and q on the toy dataset shown below. Let $x \in \mathbb{R}^2$, where each point has 2 features x_1 and x_2. In the plot below, the pluses are positive points and the circles are negative points. The plot shows x_1 on the horizontal axis and x_2 on the vertical axis.



(a) (4 points) Assume that $p = 1$. What is the smallest possible positive integer value of q such that the model can learn a decision boundary that perfectly separates the positive and negative datapoints? Explain your reasoning.

Solution: The answer is 2. With $q = 2$, we can have all quadratic functions, which is enough to separate the given points (e.g. by drawing a circular decision boundary). In contrast, if $q = 1$, then the function learned is linear so it cannot separate the positive and negative examples.

(b) (7 points) Suppose that $q = 1$. **Prove** that for any value of p, the decision boundary learned by kernel logistic regression will be a straight line.

Solution: The decision boundary satisfies the equation

$$\sum_{i=1}^{n} \alpha_i k(x^{(i)}, x) = 0.$$

Plugging in this value for k, we have

$$0 = \sum_{i=1}^{n} \alpha_i (x^{(i)\top} x + p) = \left(\sum_{i=1}^{n} \alpha_i x^{(i)} \right)^\top x + p \sum_{i=1}^{n} \alpha_i.$$

This is clearly a linear function of x, so this boundary is linear.

A second solution is to notice that the kernel $k(x, z) = x^\top z + p$ corresponds to the feature function

$$\phi(x) = \begin{pmatrix} x_1 \\ x_2 \\ \sqrt{p} \end{pmatrix}.$$

The learned decision boundary is always a linear function of $\phi(x)$, which is a linear function of x, so the decision boundary is linear.

(c) (4 points) Is there a value of $q > 1000$ such that it is impossible to perfectly classify the four datapoints in the figure using kernelized linear regression? Explain your reasoning.

Solution: No. Having a higher-degree polynomial kernel can still separate the points that are linearly separable by a circle when $q = 2$, as it just adds even more features.

(d) (4 points) In general when using kernel methods, provide one potential drawback of using a very large value of q (e.g., $q = 1000$). Explain your reasoning.

Solution: This may not always be a good idea, as the boundary may be too complicated such that it may overfit to the data.

Note that computational efficiency is not a valid reason. Due to the krenel trick, using $q = 1000$ is not any slower than using $q = 3$.

(e) (4 points) Finally, let's see how other classification methods would perform on this same dataset. Suppose we ran 1-nearest neighbor using 4-fold cross-validation (i.e., each example is in its own fold). What would be the validation accuracy? Show your work. Use the Euclidean distance to compute nearest neighbors.

Solution: We can see from the figure that for each example, its nearest neighbor is one of the opposite label. Thus, the 4-fold cross-validation accuracy is in fact 0.

Question 3: RNN and MLP Parameters (24 points)

In this question, you will help Lorena and Bill set up and train neural network models for text data.

(a) Lorena wants to use an Elman RNN, whose recurrence is defined by the equation:

$$h_t = \tanh(W^h h_{t-1} + W^x x_t + b).$$

She decides to use 128-dimensional hidden states and 50-dimensional word vectors, have a vocabulary of size 1000, and have a dataset where each example can have up to 200 words. Compute the following:

i. (2 points) The dimension the W^h

> **Solution:** 128×128.

ii. (2 points) The dimension of W^x

> **Solution:** 128×50

iii. (2 points) The total number of parameters for all word vectors.

> **Solution:** $1000 \times 50 = 50,000$.

(b) (4 points) Explain why a function like tanh is used in the Elman RNN update formula. What would happen if the tanh were removed?

> **Solution:** It is important to have a non-linearity so that the model can extract more complicated non-linear features. Without the tanh, the hidden state would simply be a linear function of all the word vectors, which is much less expressive.

(c) (6 points) Lorena now wants to train her RNN to generate text. In particular:

- She has a training document with words $u_1, u_2, \ldots, u_T, u_{T+1}$. u_1 is the special "beginning of sequence" token `[BEGIN]`, and u_{T+1} is the special "end of sequence" token `[END]`. For example, if her document was "Fight on," then we would have $T = 3$ and $u_1 = $ [`BEGIN`], $u_2 = $ Fight, $u_3 = $ on, and $u_4 = $ [`END`].
- During training, she feeds this sequence of words to the RNN one at a time (i.e., she uses teacher forcing), except for the final `[END]` token. This generates a sequence of hidden states $h_1, \ldots, h_T$.
- The hidden states are fed directly into a softmax regression-type layer to predict the next word. This layer has a separate parameter vector $w^{(u)}$ for each word u in the vocabulary V.
- Lorena computes the loss on this document by summing up the losses for generating each "next word" from u_2 to u_{T+1}.

Write the mathematical expression for the training loss Lorena computes on this training document. It may help to remember that in softmax regression, given input x and C weight vectors $w^{(1)}, \ldots, w^{(C)}$, the probability that $y = j$ is given by

$$P(y = j \mid x) = \frac{\exp(w^{(j)\top} x)}{\sum_{k=1}^{C} \exp(w^{(k)\top} x)}.$$

Solution: The MLE loss is:

$$-\sum_{t=1}^{T}\left(w^{(u_{t+1})\top}h_t - \log\sum_{v\in V}\exp(w^{(v)\top}h_t)\right).$$

An equivalent way to write this is

$$-\sum_{t=2}^{T+1}\left(w^{(u_t)\top}h_{t-1} - \log\sum_{v\in V}\exp(w^{(v)\top}h_{t-1})\right).$$

Although a slight abuse of notation, it was also acceptable to use a sum over an index from 1 to $|V|$ instead of a sum over elements of V. (Technically the problem asks you to index into w's using words, not using integers, but this was close enough to get full credit.)

(d) (4 points) Bill doesn't like RNNs, so he comes up with an idea to use an MLP instead. Since he anticipates the maximum length of a document to be 200 words, he plans to build a model that can take in the concatenation of 200 word vectors as input. (If a document has less than 200 words, he will concatenate enough zero vectors to make it the same length as 200 word vectors.) He will use 50-dimensional word vectors, and an MLP with 500 hidden units. How many parameters will he need in the first layer of his MLP? Note: Include both the weight matrix and bias vector.

Solution: The input is $200 \times 50 = 10,000$-dimensional, and the output is 500-dimensional. So, the weight matrix will be $500 \times 10,000$, and the bias will be another 500. The total number of parameters is therefore $500 \times 10,000 + 500 = \boxed{5,000,500}$.

(e) (4 points) Using the terms "bias" and/or "variance," explain why having a large number of parameters in your model can make it harder to have high test accuracy.

Solution: When you have more parameters, you will have higher variance because there will be more ways to fit your available data, and it's harder to identify the optimal choice of parameters. This can lead to overfitting.

Question 4: Short Response (12 points)

Answer the following questions and **explain your reasoning fully**. You may also draw explanatory diagrams when appropriate.

(a) (4 points) Is the function $f(x) = \max(x, 0)$ from $\mathbb{R} \to \mathbb{R}$ a convex function?

Solution: Yes this is convex. You can show this by sketching the function and showing that all lines connecting two points lie above the function.

Note that you **cannot** show this by taking the second derivative and showing that it is ≥ 0 everywhere it is defined. The second derivative is not defined everywhere, since $f(x)$ is not differentiable at $x = 0$, so the second derivative test is not applicable.

Consider how the function $f(x) = -\max(x, 0)$ also has a second derivative that is ≥ 0 everywhere it is defined, but is not convex.

(b) (4 points) When training a neural network using stochastic gradient descent, is the loss on the training dataset guaranteed to decrease at every iteration?

Solution: No. It depends on the specific data points that are chosen for updating the parameters during that iteration. The objective loss function may be increased due to the direction of the gradient with respect to the chosen data points.

(c) (4 points) Generative classifiers make predictions by estimating $P(y)$ and $P(x \mid y)$. Is the Naive Bayes assumption used when modeling $P(y)$, $P(x \mid y)$, or both?

Solution: The Naive Bayes assumption is specifically about $P(x \mid y)$ being the product of independent distributions. It is not used when modeling $P(y)$.

Question 5: Multiple Choice (13 points)

In the following questions, circle the correct answer(s). There is no need to explain your answer.

(a) (2 points) Suppose we are training a neural network with SGD using a batch size of 50, on a training dataset with 50,000 examples. How many total gradient updates would be performed after 5 epochs?

A. 1,000
B. 5,000
C. 50,000
D. 250,000

Solution: B. Each epoch performs $50,000/50 = 1,000$ updates, so 5 epochs is $5,000$ updates.

(b) (2 points) Which of the following situations is called overfitting?

A. Low training error, low test error
B. High training error, high test error
C. High training error, low test error
D. Low training error, high test error

Solution: D.

(c) (3 points) Consider a training set $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})$ and a probabilistic model $\Pr(y \mid x; w)$ which specifies the probability of seeing outcome y given x and parameterized by w. Which of the following is the value of w chosen by Maximum Likelihood Estimation (MLE)? Choose all that apply.

A. $\arg\max_w \sum_{i=1}^{n} P(y^{(i)} \mid x^{(i)}; w)$
B. $\arg\max_w \prod_{i=1}^{n} P(y^{(i)} \mid x^{(i)}; w)$
C. $\arg\max_w \sum_{i=1}^{n} \log P(y^{(i)} \mid x^{(i)}; w)$

D. $\arg\max_w \prod_{i=1}^n \log P(y^{(i)} \mid x^{(i)}; w)$

Solution: B and C are both correct. Note that B is simply the log of C.

(d) (3 points) Which of the following would **not** be a valid loss function for linear regression? Choose all that apply.

A. $\mathcal{L}(w) = \frac{1}{n}\sum_{i=1}^n (w^\top x^{(i)} - y^{(i)})^3$
B. $\mathcal{L}(w) = \frac{1}{n}\sum_{i=1}^n (w^\top x^{(i)} - y^{(i)})^2$
C. $\mathcal{L}(w) = \frac{1}{n}\sum_{i=1}^n |w^\top x^{(i)} - y^{(i)}|$
D. $\mathcal{L}(w) = \frac{1}{n}\sum_{i=1}^n (w^\top x^{(i)} - y^{(i)})$

Solution: A and D. They are both bad because they give you very low (i.e., negative) loss for under-predicting the correct answer. B is the normal linear regression and C is the Laplacian regression from the homework.

(e) (3 points) Which of the following strategies may help address overfitting? Choose all that apply.

A. Acquire more training data.
B. Train the model with a small subset of the training dataset.
C. Apply regularization on the parameters.
D. Remove regularization on the parameters.

Solution: A and C. More training data helps you identify the optimal parameters, and regularization is used to prevent overfitting by definition.

This content is protected and may not be shared, uploaded, or distributed.