ELEC 378 — Machine Learning: Concepts & Techniques
Final Exam
Take-Home Due: 11:59pm May 2, 2023

INSTRUCTIONS

1. This exam is CLOSED BOOK, CLOSED NOTES, and CLOSED ANY OTHER RESOURCE (including calculators and computers), except that you are allowed TWO $8\frac{1}{2} \times 11$ inch sheets of notes (both sides).

2. You have 3 hours to complete the exam; take it at one sitting. You should write in an exam booklet or on sheets of blank paper.

3. This test is to be completed on your own — NO COLLABORATION WITH OTHERS ALLOWED.

4. Include your note sheets at the end of your test. Minus 10 points if you don't include them.

5. Write clearly; if we can't read it, you won't get credit. Show your work.

6. Sign the pledge when you are finished.

7. **Submit the test on Gradescope.** Late tests will *not* be accepted.

NOTATION

- A **training data set** for supervised learning consists of n labeled data points $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$ with each $\mathbf{x}_i \in \mathbb{R}^p$.
- A **training data set** for unsupervised learning consists of n unlabeled data points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ with each $\mathbf{x}_i \in \mathbb{R}^p$.

1. **QUIKKIES**$^{\text{TM}}$ (5 points each; 40 points total)

   (a) What's the difference between regression and classification, **and** why do we approach these two similar problems with different methods?

   (b) Suppose the optimal parameters of an SVM are learned via gradient descent to be $\mathbf{w} = \begin{bmatrix} 6 & 3 \end{bmatrix}^\intercal$ and $b = -9$, where $\hat{y}_i = \text{sign}(\langle \mathbf{w}, \mathbf{x}_i \rangle + b)$ is the SVM's prediction of the label y_i of data point $\mathbf{x}_i \in \mathbb{R}^2$. What are the slope and intercept of the hyperplane parameterized by $\mathbf{w}$ and b? Plot the hyperplane in $\mathbb{R}^2$ along with a possible dataset from which this hyperplane could be learned.

   (c) What does it mean to "do PCA"? Describe explicitly all necessary computations, and explain how and why PCA can be used for dimensionality reduction.

   (d) The binary SVM requires each data point to have one of two possible labels, $y_i \in \{+1, -1\}$. How can you apply such a binary model to implement multiclass classification, where each data point has one of $K > 2$ arbitrary labels, $y_i \in \{c_1, \ldots, c_K\}$?

   (e) How can you use a linear classifier to learn the decision boundary of a binary labeled dataset whose two classes are nonlinearly separable?

   (f) Both SVM and logistic regression learn a hyperplane with parameters $\mathbf{w}$ and b to make predictions of the form $\hat{y}_i = \sigma(\langle \mathbf{w}, \mathbf{x}_i \rangle + b)$, where $\mathbf{x}_i$ is a data point, σ is some non-linear function, and $\mathbf{w}$ and b are learned from training data so that the prediction $\hat{y}_i$ is close to the ground truth label y_i. What is σ in SVM? What is σ in logistic regression? And what is one key difference between the hyperplanes learned by these two linear classifiers?

   (g) What is the purpose of cross validation? Describe a scenario in which you would use it and how it is implemented in practice.

   (h) Both the kernel SVM and deep neural network models can learn nonlinear decision boundaries. When should you choose a kernel SVM over a deep neural network, and vice versa?

2. **Playing with Perceptrons** (25 points)

In each part of this problem, consider the labeled training data set

$$\begin{aligned}\mathbf{x}_1 &= \begin{bmatrix}0 & 0\end{bmatrix}^\intercal,\ y_1 = 0\\ \mathbf{x}_2 &= \begin{bmatrix}1 & 0\end{bmatrix}^\intercal,\ y_2 = 1\\ \mathbf{x}_3 &= \begin{bmatrix}0 & 1\end{bmatrix}^\intercal,\ y_3 = 1\\ \mathbf{x}_4 &= \begin{bmatrix}1 & 1\end{bmatrix}^\intercal,\ y_4 = 0.\end{aligned}$$

(a) Plot the training data $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4$ in $\mathbb{R}^2$, and label each data point $\mathbf{x}_i$ with its label y_i. What is n? What is p?

(b) Is it possible to find parameters $\mathbf{w}$, b and nonlinearity σ so that a single layer perceptron prediction $\hat{y}_i = \sigma(\langle \mathbf{w}, \mathbf{x}_i \rangle + b)$ correctly predicts the label of each of the data points $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4$? If so, provide the corresponding $\mathbf{w}$, b, and σ. If not, justify why.

(c) Let $\phi(\mathbf{x}_i) = (x_i[1] - 1/2)(x_i[2] - 1/2) \in \mathbb{R}$, where $\mathbf{x}_i = \begin{bmatrix}x_i[1] & x_i[2]\end{bmatrix}^\intercal$. Plot the *transformed training data* $\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), \phi(\mathbf{x}_3), \phi(\mathbf{x}_4)$ in $\mathbb{R}$ and label each transformed data point $\phi(\mathbf{x}_i)$ with its (unchanged) label y_i. What are n and p in the *transformed training data*?

(d) Is $\phi(\mathbf{x}_i)$ a *linear* or *nonlinear* transform of the data point $\mathbf{x}_i$? Justify your answer with a proof.

(e) Is it possible to find parameters v, c and nonlinearity σ so that a single layer perceptron prediction $\hat{z}_i = \sigma(v \cdot \phi(\mathbf{x}_i) + c)$ correctly predicts the label of each of the *transformed data points* $\phi(\mathbf{x}_1), \phi(\mathbf{x}_2), \phi(\mathbf{x}_3), \phi(\mathbf{x}_4)$? If so, provide the corresponding v, c, and σ. If not, justify why.

3. **Inner Product Classifier** (35 points)

Let $\{\mathbf{x}_i, y_i\}_{i=1}^n$ be labeled training data with data points $\mathbf{x}_i \in \mathbb{R}^p$ and binary labels $y_i \in \{0, 1\}$. Consider a **linear classifier** that predicts the label $\widehat{y}$ of a data point $\mathbf{x}$ as $\widehat{y} = u(\langle \mathbf{x}, \mathbf{w} \rangle - b)$, where $\mathbf{w} \in \mathbb{R}^p$ and $b \in \mathbb{R}$ are learned from the training data, $u(z)$ is the unit step function, and $\langle \cdot, \cdot \rangle$ is the inner (dot) product:

$$\begin{aligned} \widehat{y} &= u(\langle \mathbf{x}, \mathbf{w} \rangle - b) \qquad (1) \\ &= \begin{cases} 1, & \langle \mathbf{x}, \mathbf{w} \rangle - b \geq 0 \\ 0, & \langle \mathbf{x}, \mathbf{w} \rangle - b < 0 \end{cases} \\ &= \begin{cases} 1, & \langle \mathbf{x}, \mathbf{w} \rangle \geq b \\ 0, & \langle \mathbf{x}, \mathbf{w} \rangle < b \end{cases} \end{aligned}$$

(a) If this model makes the prediction $\widehat{y}_i = u(\langle \mathbf{x}_i, \mathbf{w} \rangle - b)$ for data point $\mathbf{x}_i$ with ground-truth label y_i, one can measure the model error by computing $\frac{1}{2}(y_i - \widehat{y}_i)^2$. Why may this quantity be preferred over defining the model error as $|y_i - \widehat{y}_i|$ when it comes to learning the parameters via gradient descent?

(b) The model error for the entire training data set is given by

$$\begin{aligned} \ell(\mathbf{w}, b) &= \sum_{i=1}^n \frac{1}{2}(y_i - \widehat{y}_i)^2 \\ &= \frac{1}{2} \sum_{i=1}^n (y_i - u(\langle \mathbf{x}_i, \mathbf{w} \rangle - b))^2, \end{aligned}$$

which depends on the model parameters $\mathbf{w}$ and b. In order to find parameters $\mathbf{w}$ and b that minimize the loss $\ell(\mathbf{w}, b)$ via gradient descent, why may one prefer to replace the unit step function with the sigmoid function, that is, to define $u(z) = \sigma(z) = \frac{1}{1+e^{-z}}$?

(c) On the other hand, what issues may arise in making predictions of the form $\hat{y} = \sigma(\langle \mathbf{x}, \mathbf{w} \rangle + b)$ on the training data described above when using $\sigma(z) = \frac{1}{1+e^{-z}}$?

(d) For the remaining parts, let $\hat{y} = \sigma(\langle \mathbf{x}, \mathbf{w} \rangle + b)$ be the model prediction with $\sigma(z) = \frac{1}{1+e^{-z}}$ and parameters $\mathbf{w}$ and b chosen to minimize the loss $\ell(\mathbf{w}, b) = \sum_{i=1}^n \frac{1}{2}(y_i - \widehat{y}_i)^2$. What makes this situation different than the logistic regression model?

(e) Gradient descent can be used to learn the parameters $\mathbf{w}$ and b by computing

$$\begin{aligned} \mathbf{w}_{t+1} &= \mathbf{w}_t - \mu \nabla_{\mathbf{w}} \ell(\mathbf{w}, b) \\ b_{t+1} &= b_t - \mu \nabla_b \ell(\mathbf{w}, b) \end{aligned}$$

for $t = 0, \ldots, T$, where T is a fixed integer, μ is a fixed real number, and $\mathbf{w}_0$ and b_0 are randomly chosen. One can show that

$$\frac{d}{dz}\sigma(z) = \sigma(z)(1 - \sigma(z)),$$

and thus

$$\nabla_{\mathbf{v}}\sigma(f(\mathbf{v})) = \sigma(f(\mathbf{v}))(1-\sigma(f(\mathbf{v})))\nabla_{\mathbf{v}}f(\mathbf{v})$$

for differentiable $f : \mathbb{R}^p \to \mathbb{R}$. Use this result to find expressions for $\nabla_{\mathbf{w}}\ell(\mathbf{w}, b)$ and $\nabla_b\ell(\mathbf{w}, b)$.

(f) Suppose we replace the data $\mathbf{x}_i \in \mathbb{R}^p$ with a nonlinear function $\phi(\mathbf{x}_i) \in \mathbb{R}^P$, $P \gg p$, to provide the linear classifier some nonlinear capabilities. Explain why, as defined above, it is not necessarily possible to form a prediction on the transformed data using less than $O(P)$ computations for an arbitrary nonlinear $\phi : \mathbb{R}^p \to \mathbb{R}^P$.

(g) Suppose that there exists $\boldsymbol{\alpha} \in \mathbb{R}^n$ such that $\mathbf{w} = \mathbf{X}^\intercal\boldsymbol{\alpha} = \sum_{i=1}^n \alpha_i\mathbf{x}_i$, where $\mathbf{X} = \begin{bmatrix}\mathbf{x}_1 & \cdots & \mathbf{x}_n\end{bmatrix}^\intercal$ is the data matrix. Does this assumption alone allow you to apply the so-called "kernel trick" to the nonlinear classifier proposed above, thus reducing the computational complexity of forming predictions on the transformed data $\phi(\mathbf{x}_i)$ from $O(P)$ to $O(p)$? If so, explicitly show how the prediction $\widehat{y}_i$ can be formed on the transformed data $\phi(\mathbf{x}_i)$ without explicitly transforming the data for arbitrary ϕ as defined above. If not, provide any additional assumptions on the classifier and/or the transform ϕ that are necessary for the kernel trick to work.