Homework 1

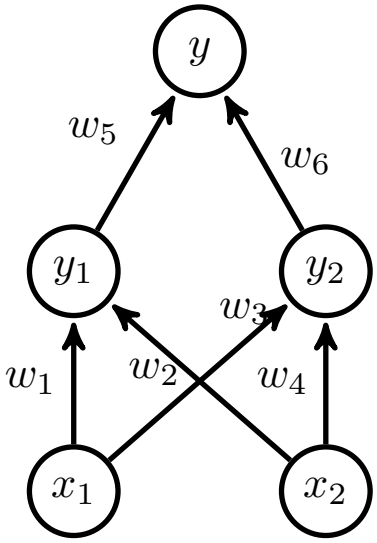
Spring 25, CS 442: Trustworthy Machine Learning
Due Friday Feb. 21st at 23:59 CT

Instructor: Han Zhao

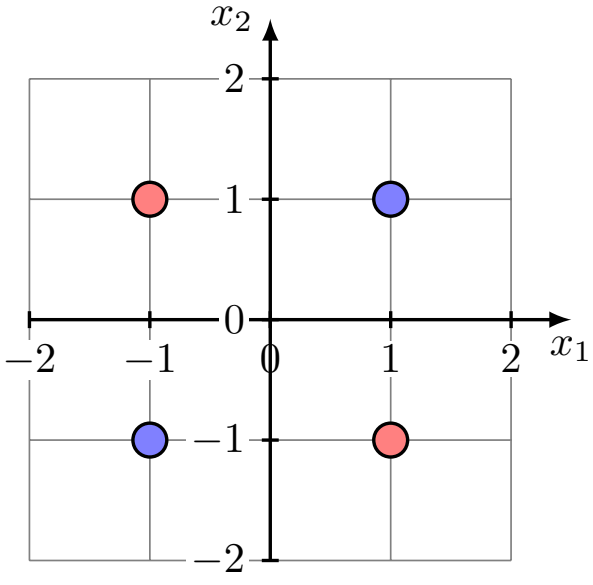
Instructions for submission All the homework submissions should be typeset in LaTeX. For all the questions, please clearly justify each step in your derivations or proofs.

1 Feed-forward Neural Networks [30 pts]

Consider a feed-forward neural network (FNN) with one input layer, one hidden layer and one output layer shown in Fig. 1a. In this problem we will use the FNN to classify the XOR pattern shown in Fig. 1b. Suppose that the activation function at every unit in the FNN are linear, i.e., $y_1 = w_1 x_1 + w_2 x_2$, $y_2 = w_3 x_1 + w_4 x_2$ and $y = w_5 y_1 + w_6 y_2 + w_0$. Classify the input instance (x_1, x_2) as $+1$ if $y(x_1, x_2) \geq 0$ otherwise -1.



(a) A three-layer feed-forward neural network with two inputs.



(b) XOR pattern. Instances in blue have label $+1$ while instances in red have label -1.

1.1 [10pts]

For any weight configuration $(w_0, w_1, w_2, w_3, w_4, w_5, w_6)$ of the FNN shown above, can you find another FNN with only two layers, i.e., one input layer with two inputs x_1, x_2 and one output layer with one output unit y that computes the same function? If yes, describe a such two-layer FNN and give the weights as functions of $(w_0, w_1, w_2, w_3, w_4, w_5, w_6)$. If no, briefly describe your reasoning.

Solution:
Yes, such a two-layer FNN exists. Since all activations are linear, the three-layer network computes:

$$y(x_1, x_2) = w_5(w_1x_1 + w_2x_2) + w_6(w_3x_1 + w_4x_2) + w_0$$

Expanding the expression:

$$y(x_1, x_2) = (w_5w_1 + w_6w_3)x_1 + (w_5w_2 + w_6w_4)x_2 + w_0$$

Thus, we can construct an equivalent two-layer FNN with one input layer and one output layer as:

$$y(x_1, x_2) = \alpha_1x_1 + \alpha_2x_2 + \alpha_0$$

where the new weights are given by:

$$\alpha_1 = w_5w_1 + w_6w_3, \quad \alpha_2 = w_5w_2 + w_6w_4, \quad \alpha_0 = w_0$$

This confirms that the original three-layer network can be collapsed into a two-layer network with appropriately transformed weights.

1.2 [10pts]

Show that there is no weight configuration $(w_0, w_1, w_2, w_3, w_4, w_5, w_6)$ under which the FNN shown in Fig. 1a can classify the XOR pattern with no error.

Solution:
The mapping in 1.1 is linear. However, the XOR pattern is not linearly separable. No linear function can correctly assign one class to $(x_1, x_2) = (0, 0)$ and $(1, 1)$ while assigning the opposite class to $(0, 1)$ and $(1, 0)$.
For any choice of weights $(w_0, \ldots, w_6)$ the network is equivalent to a linear classifier and cannot perfectly classify the XOR pattern.

1.3 [10pts]

For the FNN shown in Fig. 1a, will changing the activation functions at y_1 and y_2 to be nonlinear help classify the XOR pattern? If yes, construct a nonlinear activation function $f(\cdot)$ to be applied only at y_1 and y_2, i.e., $y_1(x_1, x_2) = f(w_1x_1 + w_2x_2)$, $y_2(x_1, x_2) = f(w_3x_1 + w_4x_2)$, such that the new FNN can perfectly classify the XOR pattern. If no, briefly describe your reasoning on why it is not possible.

Solution:

Yes. The key issue in 1.2 is that linear functions cannot express the required nonlinearity. By introducing a nonlinear activation function $f(\cdot)$ at the hidden units, the network can "bend" the decision boundary. For example, define the activation function f by:

$$f(z) = \begin{cases} 1, & \text{if } z \geq 0 \\ 0, & \text{if } z < 0 \end{cases}$$

Now choose the hidden layer weights and biases as follows:
- For the first hidden unit, let $z_1 = x_1 + x_2 - 0.5$ so that:

$$y_1 = f(x_1 + x_2 - 0.5).$$

- For the second hidden unit, let $z_2 = x_1 + x_2 - 1.5$ so that:

$$y_2 = f(x_1 + x_2 - 1.5).$$

Then, define the output unit as:

$$y = y_1 - y_2$$

and use the rule "predict $+1$ if $y > 0$, and -1 otherwise."

Now check the four cases (interpreting the XOR pattern with inputs in $\{0,1\}$ and labels defined as follows: $(0,0)$ and $(1,1)$ should yield -1, while $(0,1)$ and $(1,0)$ should yield $+1$):

- $(0,0)$: $y_1 = f(-0.5) = 0$, $y_2 = f(-1.5) = 0$, $y = 0 - 0 = 0 \Rightarrow$ classify -1.
- $(0,1)$ or $(1,0)$: $y_1 = f(0.5) = 1$, $y_2 = f(-0.5) = 0$, $y = 1 - 0 = 1 \Rightarrow$ classify $+1$.
- $(1,1)$: $y_1 = f(1.5) = 1$, $y_2 = f(0.5) = 1$, $y = 1 - 1 = 0 \Rightarrow$ classify -1.

This shows that with a proper nonlinear activation, the FNN can perfectly classify the XOR pattern.

2 The Price of Statistical Parity [30pts]

Consider a binary classification problem with two groups. Let (X, A, Y) be the tuple drawn from an underlying distribution μ. For simplicity, in this problem we assume all the variables are binary, i.e., $X, A, Y \in \{0,1\}$. Recall from the lecture, in this example we use A to denote the group membership of an instance, i.e., $A = 0$ means the majority group whereas $A = 1$ means the minority group. More concretely, let $p \geq 1/2$ be the marginal probability of $A = 0$: $\Pr_\mu(A = 0) = p$. To complete the specification of the joint distribution μ over (X, A, Y), the group-wise distributions over the pair (X, Y) are given as follows:

- For each group $A = a \in \{0,1\}$, the conditional probability $\Pr_\mu(X = 1 \mid A = a) = 1/2$ is uniform, i.e., with equal probabilities, X can take either 0 or 1.
- For each group $A = a \in \{0,1\}$, $\Pr_\mu(Y = a \mid A = a) = 1$, i.e., with probability 1 the value of the target Y equals a.

2.1 [10pts]

Show that there exists a deterministic classifier $h : \{0,1\} \times \{0,1\} \to \{0,1\}$, taking the pair (x, a) as input and predicts the corresponding label, is simultaneously perfect on both groups. Construct such a deterministic classifier. Note: we say a classifier to be perfect if it achieves 0 classification error on the corresponding distribution.

Solution:
The following deterministic classifier is perfect:

$$h(x, a) = a.$$

For any input (x, a), this classifier outputs a, which exactly matches Y.

2.2 [10pts]

Now we consider a randomized classifier h as follows. Upon receiving the input pair (x, a), a randomized classifier $h(x, a)$ will flip a fair coin. Depending on the outcome of the coin, the randomized classifier $h(x, a)$ will make the following prediction:

$$h(x, a) = \begin{cases} 0 & \text{If the outcome of the fair coin is H} \\ 1 & \text{If the outcome of the fair coin is T.} \end{cases}$$

2.2.1 [5pts]

What is the classification error rate of this randomized classifier on the joint distribution μ? i.e., what is $\Pr_{(X,A,Y)\sim\mu}(h(X, A) \neq Y)$?

Solution:
For group $A = 0$ (where $Y = 0$), the classifier is correct if it outputs 0 with probability $\frac{1}{2}$. For group $A = 1$ (where $Y = 1$), it is correct if it outputs 1 with probability $\frac{1}{2}$.
Therefore, the classification error is $\frac{1}{2}$.

2.2.2 [5pts]

Show that despite taking the group membership A explicitly as its input, the above randomized classifier satisfies statistical parity.

Solution:
A classifier satisfies statistical parity if the probability of predicting a certain outcome is the same across groups. Here, regardless of A, the probability of predicting 1 is $\frac{1}{2}$ and predicting 0 is $\frac{1}{2}$.
Thus, despite taking the group membership A explicitly as an input, the output distribution does not depend on A. Hence, the classifier satisfies statistical parity.

2.3 [10pts]

2.3.1 [8pts]

Prove that, for any deterministic classifier $h(X, A)$[1], if $h(X, A)$ satisfies statistical parity, then

$$\Pr_{\mu}(h(X, A) \neq Y \mid A = 0) + \Pr_{\mu}(h(X, A) \neq Y \mid A = 1) = 1.$$

Solution:
Assume $h(x, a)$ is a deterministic classifier that satisfies *statistical parity*. Statistical parity implies that

$$\Pr[h(X, 0) = 1] = \Pr[h(X, 1) = 1] =: q.$$

Thus,

$$\Pr_{\mu}(h(X, A) \neq Y \mid A = 0) = \Pr_{\mu}(h(X, A) = Y \mid A = 1) = q.$$

This proves that

$$\Pr_{\mu}(h(X, A) \neq Y \mid A = 0) + \Pr_{\mu}(h(X, A) \neq Y \mid A = 1) = q + (1 - q) = 1.$$

2.3.2 [2pts]

Using the result above, show that the overall error rate of h over μ is at least $1 - p$, i.e., $\Pr_{\mu}(h(X, A) \neq Y) \geq 1 - p$.

Solution:
The overall error is

$$\begin{aligned}\Pr_{\mu}(h(X, A) \neq Y) = &\Pr_{\mu}(h(X, A) \neq Y \mid A = 0) \cdot \Pr_{\mu}(A = 0) \\ &+ \Pr_{\mu}(h(X, A) \neq Y \mid A = 1) \cdot \Pr_{\mu}(A = 1).\end{aligned}$$

Substitute $\Pr_{\mu}(A = 0) = p$ and $\Pr_{\mu}(A = 1) = 1 - p$. Also, using the result from 2.3.1, substitute $\Pr_{\mu}(h(X, A) \neq Y \mid A = 1) = 1 - \Pr_{\mu}(h(X, A) \neq Y \mid A = 0)$

$$\begin{aligned}\Pr_{\mu}(h(X, A) \neq Y) &= p \cdot \Pr_{\mu}(h(X, A) \neq Y \mid A = 0) \\ &\quad + (1 - p)(1 - \Pr_{\mu}(h(X, A) \neq Y \mid A = 0)) \\ &= (2p - 1) \cdot \Pr_{\mu}(h(X, A) \neq Y \mid A = 0) + (1 - p).\end{aligned}$$

Since $p \geq \frac{1}{2}$, we have $(2p - 1) \geq 0$. Thus, the minimum overall error (achieved when $\Pr_{\mu}(h(X, A) \neq Y \mid A = 0) = 0$) is $1 - p$. Therefore,

$$\Pr_{\mu}(h(X, A) \neq Y) \geq 1 - p.$$

[1] This actually also applies to randomized classifiers as well, but in this problem you only need to show this for deterministic classifiers.

3 Matrix Calculus [10pts]

A classic problem in unsupervised machine learning is known as *symmetric matrix factorization*. Given a matrix $P \in \mathbb{R}^{n\times n}$, the goal is to find a matrix $X \in \mathbb{R}^{n\times k}$ such that $P \approx XX^\top$. In this problem, we will derive the gradient of the following objective function with respect to X:

$$\min_{X\in\mathbb{R}^{n\times k}} \quad \mathcal{L}(X) := \frac{1}{2}\|P - XX^\top\|_F^2,$$

where $\|\cdot\|_F$ denotes the Frobenius norm of a matrix, i.e,. $\|A\|_F^2 = \sum_{i=1}^m \sum_{j=1}^n A_{ij}^2$ for $A \in \mathbb{R}^{m\times n}$. One typical application of the above problem is to find an embedding of n objects into a k-dimensional space, where P_{ij} denotes the similarity between the i-th and j-th objects. The above objective function encourages the similarity between the i-th and j-th objects to be approximated by the inner product between the i-th and j-th rows of X.
Please derive the gradient of $\mathcal{L}(X)$ with respect to X. Note: you need to show your derivation step by step, and you need to express the gradient in matrix notation in terms of X and P only.

Solution:
We wish to compute the gradient of

$$\mathcal{L}(X) = \frac{1}{2}\|P - XX^\top\|_F^2$$

with respect to X. Define $Z(X) = P - XX^\top$, so that

$$\mathcal{L}(X) = \frac{1}{2}\|Z(X)\|_F^2.$$

The squared Frobenius norm is given by

$$\frac{1}{2}\|Z\|_F^2 = \frac{1}{2}\operatorname{Tr}(Z^\top Z).$$

Differentiate with respect to Z,

$$\nabla_Z \left[\frac{1}{2}\|Z\|_F^2\right] = \nabla_Z \left[\frac{1}{2}\operatorname{Tr}(Z^\top Z)\right] = Z.$$

Also, we can obtain

$$\frac{\partial Z}{\partial X} = \frac{\partial}{\partial X}(P - XX^\top) = -2X.$$

By the chain rule,

$$\nabla_X \mathcal{L}(X) = Z(X) \cdot \frac{\partial Z}{\partial X} = (P - XX^\top)\cdot(-2X).$$

Rearranging,

$$\nabla_X \mathcal{L}(X) = 2(XX^\top - P)X.$$

This is the gradient of $\mathcal{L}(X)$ with respect to X.

4 The Implicit Bias of Gradient Descent in Linear Regression [30 pts]

Consider a dataset $\{(x_i, y_i)\}_{i=1}^n$ where $x_i \in \mathbb{R}^d$ is the i-th input and $y_i \in \mathbb{R}$ is the i-th label. The loss function for linear regression on this dataset is given by

$$\min_{w \in \mathbb{R}^d} \quad \mathcal{L}(w) := \frac{1}{2} \sum_{i=1}^{n} (w \cdot x_i - y_i)^2, \tag{1}$$

where $w \in \mathbb{R}^d$ is the model parameter of interest. In the course we find the optimal solution w^* via solving the *normal equation*. In this problem, instead, we will use the gradient descent method to numerically find the optimal solution instead. In particular, in order to solve (1), we apply the following algorithm:

Procedure 1 Gradient Descent for Linear Regression

Input: Initial model parameter w_0

1: **for** $t = 1, 2, \ldots$ until convergence **do**
2: $\quad w_t \leftarrow w_{t-1} - \frac{1}{t} \nabla_w \mathcal{L}(w_{t-1})$
3: **end for**
4: **return** w^*

In this problem, let's assume that the above algorithm will converge, and we use w^* to denote the convergent point.

4.1 [10pts]

Prove that $w^* - w_0 \in \text{span}\{x_1, \ldots, x_n\}$.

Solution:

Notice that the gradient at any step is given by

$$\nabla L(w) = \frac{1}{n} \sum_i (w \cdot x_i - y_i) x_i,$$

which is clearly a linear combination of the training inputs $\{x_1, \ldots, x_n\}$. Starting from w_0, the first update is

$$w_1 = w_0 - (1) \cdot (\text{linear combination of } x_i),$$

so that $w_1 - w_0$ lies in $\text{span}\{x_1, \ldots, x_n\}$.
Suppose by induction that $w_t - w_0 \in \text{span}\{x_1, \ldots, x_n\}$. Then the update

$$w_{t+1} = w_t - \frac{1}{t+1} \nabla L(w_t)$$

implies that

$$w_{t+1} - w_0 = (w_t - w_0) - \frac{1}{t+1} \nabla L(w_t),$$

which is a sum of vectors in $\text{span}\{x_1, \ldots, x_n\}$. Hence, by induction,

$$w^* - w_0 \in \text{span}\{x_1, \ldots, x_n\}.$$

4.2 [10pts]

Let $X \in \mathbb{R}^{n \times d}$ be the data matrix where the i-th row of this matrix is given by $x_i^\top$, and let $y = (y_1, \ldots, y_n)^\top \in \mathbb{R}^n$ be the label vector.

4.2.1 [5pts]

Show that w^* satisfies the normal equation, i.e., $X^\top X w^* = X^\top y$.

Solution:

At convergence, w^* is a stationary point so that

$$\nabla L(w^*) = \frac{1}{n} X^\top (X w^* - y) = 0.$$

Multiplying by n gives

$$X^\top (X w^* - y) = 0 \quad \Rightarrow \quad X^\top X w^* = X^\top y.$$

Thus, w^* satisfies the normal equation.

4.2.2 [5pts]

For any vector $\widehat{w} \in \mathbb{R}^d$ that is the optimal solution of (1), show that $w^* - \widehat{w} \in \text{Ker}(X)$, where $\text{Ker}(\cdot)$ denotes the kernel of a matrix.

Solution:
Let $\hat{w}$ be any optimal solution of $L(w)$. Then $\hat{w}$ satisfies the normal equation:

$$X^\top X \hat{w} = X^\top y.$$

Subtracting the normal equations for w^* and $\hat{w}$ yields

$$X^\top X (w^* - \hat{w}) = 0.$$

This implies that $w^* - \hat{w}$ is in the null space (kernel) of $X^\top X$. Because for any matrix X, $\text{Ker}(X^\top X) = \text{Ker}(X)$, we conclude that

$$w^* - \hat{w} \in \text{Ker}(X).$$

4.3 [10pts]

Prove that among all the model parameters that optimize (1), w^* has the minimum distance to w_0. Formally, let $W := \{\widehat{w} \in \mathbb{R}^d : \widehat{w} \text{ is an optimal solution to (1)}\}$. Prove that $w^* = \arg\min_{w \in W} \|w - w_0\|_2$. Note: this means that gradient descent finds the optimal solution that is closest to the initial point.

Solution:
Since 4.1 shows that $w^* - w_0$ lies in $\text{span}\{x_1, \ldots, x_n\}$ (i.e., the row space of X), and 4.2.2 shows that

for any other optimal solution $\hat{w}$, the difference $\hat{w} - w^*$ lies in $\mathrm{Ker}(X)$ (the orthogonal complement of the row space of X), we have the orthogonal decomposition:

$$\hat{w} - w_0 = (w^* - w_0) + (\hat{w} - w^*),$$

with $(w^* - w_0) \perp (\hat{w} - w^*)$.
By the Pythagorean theorem,

$$\|\hat{w} - w_0\|^2 = \|w^* - w_0\|^2 + \|\hat{w} - w^*\|^2.$$

Thus, among all $\hat{w}$ satisfying the normal equation, the one with minimum distance to w_0 is obtained by taking $\hat{w} = w^*$, since any nonzero deviation in $\mathrm{Ker}(X)$ increases the norm. In other words,

$$w^* = \arg\min_{w \in W} \|w - w_0\|,$$

which shows that gradient descent converges to the optimal solution closest to the initialization.