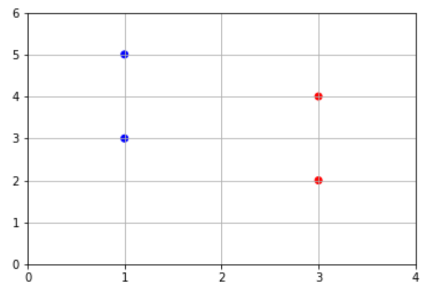
CS 182, PROBLEM SET 5

Due: November 30, 2021 11:59pm

This problem set covers Lectures 16, 17, 18. The topics include Decision Trees, Linear Classification, Neural Networks.

1. (20 points) *Comprehension.*

(1) (10 points) *Margins.* Consider the following four points. The blue points are true positive examples, and the red points are true negative examples.



(a) (5 points) What are the weights w_2, w_1, w_0 for the maximum margin separator? Note that w_0 is is called a *bias* term. The decision boundary equation for the separator should be of the form:

$$w_2 x_2 + w_1 x_1 + w_0 = 0.$$

(b) (5 points) If you scaled this separator by a positive constant k (i.e., replace w_2, w_1, w_0 with kw_2, kw_1, kw_0), would it still be a maximum margin separator?

Date: November 17, 2021.

(2) (10 points) *Neural Networks.* Consider the following input "image":

1	2	1
0	2	1
1	2	1

Suppose we run this input through a forward pass of a simple convolutional neural network. The first layer is a convolution with kernel

1	-1
1	-1

with stride 1 and padding of 1 on all four sides. The second layer is an average pooling with 2×2 filters and stride 2. What is the output of these two layers?

2. (25 points) *Decision Trees.*

(1) (15 points) Consider the following dataset comprised of three binary input attributes A, B, and C, and one binary output. Use the algorithm Learn-DT to learn a decision tree for this data. Show the computations made to determine the attribute to split at each node.

Example	Attribute A	Attribute B	Attribute C	Output
x_1	1	0	0	0
x_2	1	0	1	0
x_3	0	1	0	0
x_4	1	1	1	1
x_5	1	1	0	1

(2) (10 points) A decision *graph* is a generalization of a decision tree that allows nodes (i.e., attributes used for splits) to have multiple parents, rather than just a single parent. The resulting graph must still be acyclic. Now consider the *XOR* (or *parity*) function of *three* binary input attributes, which produces the value 1 if and only if an odd number of the three input attributes has value 1.

(a) (5 points) Draw a minimal-sized decision *tree* for the three-input XOR function.

(b) (5 points) Draw a minimal-sized decision *graph* for the three-input XOR function.

3. (25 points) *Linear Classification.*

(1) (5 points) *Higher Dimensions.* Let's figure out how to linearly separate the interior of an arbitrary circle or ellipse from its surroundings. The general equation for a circle (and hence the decision boundary) is $(x_1 - a)^2 + (x_2 - b)^2 - r^2 = 0$, and the general equation for an ellipse is $c(x_1 - a)^2 + d(x_2 - b)^2 - 1 = 0$.

(a) (2 points) Expand out the equation for the circle and show what the weights w_i would be for the decision boundary in the four-dimensional feature space x_1, x_2, x_1^2, x_2^2. Explain why this means any circle is linearly separable in this space.

(b) (3 points) Do the same for ellipses in the same feature space x_1, x_2, x_1^2, x_2^2.

(2) (10 points) *Logistic Regression Practice.* In this problem, you'll use the `sklearn` Python library to implement a logistic regression on a toy dataset. Suppose you have the following 7 examples (formatted as $((x_1, x_2), y)$):

$$((1,0),1)$$
$$((1,2),1)$$
$$((2,1),0)$$
$$((2,2),0)$$
$$((2,3),1)$$
$$((3,3),0)$$
$$((3,1),0)$$

Plot this dataset with red points for $y = 0$ and blue points for $y = 1$. Then run a logistic regression using the features x_1 and x_2 and plot the resulting decision boundary. What are the coefficients and the intercept? Include your code in this pdf file.

(3) (10 points) *Perceptron.* Consider adding a *bias term* $x_0 = 1$ to all the data points. Implement the Perceptron algorithm with initial weight $\vec{w} = (1, 1, 0)$. (Hint: As per the Perceptron conventions, you may want to redefine y to take on values of $\{-1, +1\}$ as opposed to $\{0, 1\}$.) Please include your code in this pdf file and write down how many passes through the data it took for your algorithm to converge. Does your algorithm converge when you add the point $((x_1, x_2), y) = ((2.5, 0), 1)$ to you dataset? Why or why not?

4. (30 points) *Neural Network Walkthroughs.* In this problem, you will get practice training neural network models on the MNIST dataset using different machine learning libraries. You will implement the `train` and `predict` functions using the scikit-learn, PyTorch, and TensorFlow ($\geq$ 2.0) libraries, following the instructions in file given. The autograder will verify that you obtain at least 0.95 accuracy on the test dataset. We will have 3 separate autograders you will need to submit your file to, one for each library.