CS 5350/6350: Machine Learning Fall 2022

Homework 3

Handed out: 21 Oct, 2022
Due date: 11:59pm, 4 Nov, 2022

- You are welcome to talk to other members of the class about the homework. I am more concerned that you understand the underlying concepts. However, you should write down your own solution. Please keep the class collaboration policy in mind.
- Feel free to discuss the homework with the instructor or the TAs.
- Your written solutions should be brief and clear. You do not need to include original problem descriptions in your solutions. You need to show your work, not just the final answer, but you do *not* need to write it in gory detail. Your assignment should be **no more than 15 pages**. Every extra page will cost a point.
- Handwritten solutions will not be accepted.
- *Your code should run on the CADE machines.* **You should include a shell script, `run.sh`, that will execute your code in the CADE environment. Your code should produce similar output to what you include in your report.**

  You are responsible for ensuring that the grader can execute the code using only the included script. If you are using an esoteric programming language, you should make sure that its runtime is available on CADE.
- Please do not hand in binary files! We will *not* grade binary submissions.
- The homework is due by **midnight of the due date**. Please submit the homework on Canvas.

1 Paper Problems [36 points + 15 bonus]

1. [8 points] Suppose we have a linear classifier for 2 dimensional features. The classification boundary, i.e., the hyperplane is $2x_1 + 3x_2 - 4 = 0$ (x_1 and x_2 are the two input features).

x_1	x_2	label
1	1	1
1	-1	-1
0	0	-1
-1	3	1

Table 1: Dataset 1

(a) [4 points] Now we have a dataset in Table 1. Does the hyperplane have a margin for the dataset? If yes, what is the margin? Please use the formula we discussed in the class to compute. If no, why? (Hint: when can a hyperplane have a margin?)

This hyperplane does have a margin on the data set because the data set is linearly separable with the hyperplane. We can show this by calculating the label for each point using the hyperplane.
$2(1) + 3(1) - 4 = 1$ - label is 1 so this is correct
$2(1) + 3(-1) - 4 = -5$ - label is -1 so this is correct
$2(0) + 3(0) - 4 = -4$ - label is -1 so this is correct
$2(-1) + 3(3) - 4 = 3$ - label is 1 so this is correct
To compute the margin of the hyperplane on this dataset can be computed by finding the minimum distance from each point to the hyperplane
$d(\mathbf{x}_1, h) = \frac{|(2,3,-4)^T(1,1,1)|}{||(2,3,-4)||} = \frac{1}{5.385} = 0.186$
$d(\mathbf{x}_2, h) = \frac{|(2,3,-4)^T(1,-1,1)|}{||(2,3,-4)||} = \frac{5}{5.385} = 0.928$
$d(\mathbf{x}_3, h) = \frac{|(2,3,-4)^T(0,0,1)|}{||(2,3,-4)||} = \frac{4}{5.385} = 0.743$
$d(\mathbf{x}_4, h) = \frac{|(2,3,-4)^T(-1,3,1)|}{||(2,3,-4)||} = \frac{3}{5.385} = 0.557$
The minimum distance from a point to the hyperplane was 0.186 so this is the margin of the hyperplane for this dataset.

x_1	x_2	label
1	1	1
1	-1	-1
0	0	-1
-1	3	1
-1	-1	1

Table 2: Dataset 2

(b) [4 points] We have a second dataset in Table 2. Does the hyperplane have a margin for the dataset? If yes, what is the margin? If no, why?

The hyperplane does not have a margin for this dataset because the dataset is not linearly separable by this hyperplane. We can see this by computing the last point in this dataset with the hyperplane:
$2(-1) + 3(-1) - 4 = -9$ - label is 1 so this is incorrect
Because the hyperplane can not separate all of the points in this dataset, the hyperplane does not have a margin for the dataset.

2. [8 points] Now, let us look at margins for datasets. Please review what we have discussed in the lecture and slides. A margin for a dataset is not a margin of a hyperplane!

(a) [4 points] Given the dataset in Table 3, can you calculate its margin? If you cannot, please explain why.

We can calculate the margin for this dataset because it is linearly seperable. The hyperplane that seperates this dataset with maximum margin is $x + y = 0$. Using

x_1	x_2	label
-1	0	-1
0	-1	-1
1	0	1
0	1	1

Table 3: Dataset 3

this hyperplane we can compute the margin of the dataset by calculating the minimum distance from each point to this hyperplane.
$d(\mathbf{x}_1, h) = \frac{|(1,1,0)^T(-1,0,1)|}{||(1,1,0)||} = \frac{1}{1.414} = 0.707$
$d(\mathbf{x}_2, h) = \frac{|(1,1,0)^T(0,-1,1)|}{||(1,1,0)||} = \frac{1}{1.414} = 0.707$
$d(\mathbf{x}_3, h) = \frac{|(1,1,0)^T(1,0,1)|}{||(1,1,0)||} = \frac{1}{1.414} = 0.707$
$d(\mathbf{x}_4, h) = \frac{|(1,1,0)^T(0,1,1)|}{||(1,1,0)||} = \frac{1}{1.414} = 0.707$
The minimum distance from a point to the optimal hyperplane is 0.707 so this is the margin of the dataset.

x_1	x_2	label
-1	0	-1
0	-1	1
1	0	-1
0	1	1

Table 4: Dataset 4

(b) [4 points] Given the dataset in Table 4, can you calculate its margin? If you cannot, please explain why.

We cannot calculate the margin of this dataset because the dataset is not linearly separable.

3. [**Bonus**] [5 points] Let us review the Mistake Bound Theorem for Perceptron discussed in our lecture. If we change the second assumption to be as follows: Suppose there exists a vector $\mathbf{u} \in \mathbb{R}^n$, and a positive γ, we have for each $(\mathbf{x}_i, y_i)$ in the training data, $y_i(\mathbf{u}^\top \mathbf{x}_i) \geq \gamma$. What is the upper bound for the number of mistakes made by the Perceptron algorithm? Note that $\mathbf{u}$ is unnecessary to be a unit vector.

If the vector $\mathbf{u}$ is not necessarily a unit vector then we just need to scale γ in the mistake bound. This means that the upper bound for the number of mistakes made by the Perceptron algorithm will be:

$$\left(\frac{||\mathbf{u}||R}{\gamma}\right)^2$$

4. [10 points] We want to use Perceptron to learn a disjunction as follows,

$$f(x_1, x_2, \ldots, x_n) = \neg x_1 \vee \neg \ldots \neg x_k \vee x_{k+1} \vee \ldots \vee x_{2k} \quad \text{(note that } 2k < n\text{).}$$

The training set are all 2^n Boolean input vectors in the instance space. Please derive an upper bound of the number of mistakes made by Perceptron in learning this disjunction.

First we nee to find R, the maximum norm of all training examples. Because this is a disjunction we know that the max of all attributes will be 1 and there will be a constant feature that is also 1 so $R = \sqrt{1+n}$. We can calculate the hyperplane of this disjunction to be

$$(1-x_1)+(1-x_2)+...+(1-x_k)+x_{k+1}+x_{k+2}+...+x_{2k}-1=$$

$$-x_1-x_2-...-x_k+x_{k+1}+x_{k+2}+x_{2k}+k-1=0$$

Because there are instances that lie on this hyperplane exactly, we cannot use it to get a meaningful upper bound. If we move the hyperplane slightly by 0.5, we can calculate a non-zero margin of this separating hyperplane. This hyperplane would be

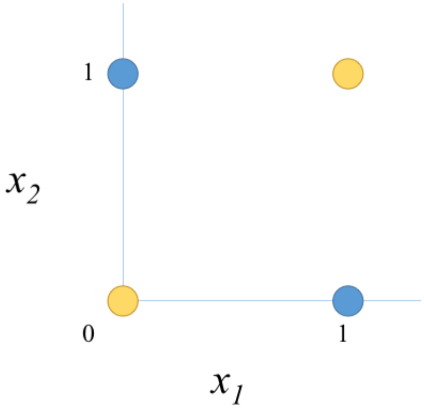
$$-x_1-x_2-...-x_k+x_{k+1}+x_{k+2}+x_{2k}+k-\frac{1}{2}=0$$

The minimum distance to this hyperplane, the margin, will be one half divided by the norm of the weight vector. The norm of the weight vector in this case would be $\sqrt{2k+k^2+\frac{1}{4}}$. Now that we know γ and R, we can compute the upper bound of the number of mistakes made by the Perceptron in learning this disjunction to be:

$$\left(\frac{\sqrt{1+n}}{\frac{\frac{1}{2}}{\sqrt{2k+k^2+\frac{1}{4}}}}\right)^2$$

5. [10 points] Prove that linear classifiers in a plane cannot shatter any 4 distinct points.

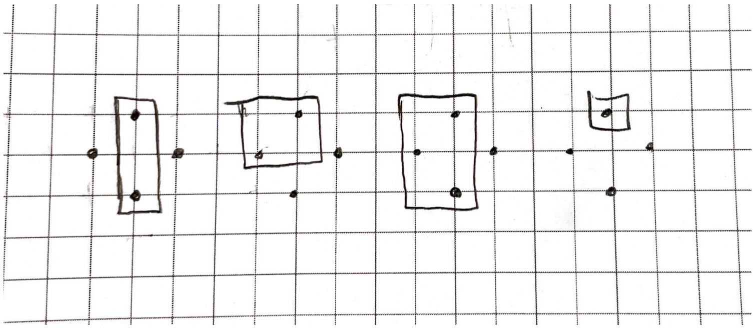
No matter how 4 points are arranged they will always either be in the form of a 4 sided polygon or in a straight line. If they are in the shape of a 4 sided polygon, there will always be a combination of plus and minus values where points along the diagonal are the same like shown in the figure:



In this case there is no line that can separate these points. If all of the points are together on a line, if the middle two points are of one class and the outside 2 are of another, this also cannot be linearly separated. Therefore linear classifiers cannot separate any 4 points.

6. [**Bonus**] [10 points] Consider our infinite hypothesis space $\mathcal{H}$ are all rectangles in a plain. Each rectangle corresponds to a classifier — all the points inside the rectangle are classified as positive, and otherwise classified as negative. What is VC($\mathcal{H}$)?

   The VC dimension of this rectangle classifier will be 4 because there exists a set of 4 points that can be shattered by this classifier but there does not exist a set of 5 points that can be shattered by this classifier.

   From the above set of points that a rectangle classifier can encapsulate any group of 1, 2, 3, or 4 of the points which means that these 4 points can be shattered by the rectangle classifier. If we add another point, the rectangle needs to be able to encapsulate all 5 points, and 4 points excluding the fifth. No matter where we add this fifth point, it will not be possible for both of these conditions to be true so this classifier cannot shatter 5 points. Therefore the VC($\mathcal{H}$) = 4

2 Practice [64 points]

1. [2 Points] Update your machine learning library. Please check in your implementation of ensemble learning and least-mean-square (LMS) method in HW1 to your GitHub repository. Remember last time you created the folders "Ensemble Learning" and "Linear Regression". You can commit your code into the corresponding folders now. Please also supplement README.md with concise descriptions about how to use your code to run your Adaboost, bagging, random forest, LMS with batch-gradient and stochastic gradient (how to call the command, set the parameters, etc). Please create a new folder "Perceptron" in the same level as these folders.

2. We will implement Perceptron for a binary classification task — bank-note authentication. Please download the data "bank-note.zip" from Canvas. The features and labels are listed in the file "bank-note/data-desc.txt". The training data are stored in the file "bank-note/train.csv", consisting of 872 examples. The test data are stored in "bank-note/test.csv", and comprise of 500 examples. In both the training and testing datasets, feature values and labels are separated by commas.

(a) [16 points] Implement the standard Perceptron. Set the maximum number of epochs T to 10. Report your learned weight vector, and the average prediction error on the test dataset.

After running the Perceptron for 10 epochs, the learned weight vector I came up with was [-61.086591, -42.70582 , -40.30786 , -3.146269, 53]. This weight vector produced an average test error of 0.02.

(b) [16 points] Implement the voted Perceptron. Set the maximum number of epochs T to 10. Report the list of the distinct weight vectors and their counts — the number of correctly predicted training examples. Using this set of weight vectors to predict each test example. Report the average test error.

The voted Perceptron came up with 259 distinct weight vectors:
w1 = [0., 0., 0., 0., 0.] - count = 1
w2 = [-3.8481, -10.1539, 3.8561, 4.2228, -1] - count = 2
w3 = [-3.800092, -8.5502, -4.6195, 3.46722 , -2] - count = 1
w4 = [-5.066792, -5.7319, -7.0455, 1.58102 , -1] - count = 8
w5 = [-7.119692, -1.8934, -7.84094, 0.36722 , 0] - count = 3
...
w255 = [-67.451391, -33.35892, -35.21686, -13.689772, 54] - count = 31
w256 = [-65.592991, -41.24492, -33.55256, -11.851372, 53] - count = 82
w257 = [-65.922191, -36.78972, -38.12436, -10.862572, 52] - count = 1
w258 = [-63.891191, -34.93772, -41.13646, -10.859569, 53] - count = 38
w259 = [-61.873491, -33.13952, -44.09456, -10.649669, 54] - count = 55

The average test error was 0.014.

(c) [16 points] Implement the average Perceptron. Set the maximum number of epochs T to 10. Report your learned weight vector. Comparing with the list of weight vectors from (b), what can you observe? Report the average prediction error on the test data.

The learned weight vector I ended up with from the averaged perceptron was: [-406358.35084101, -253467.37641, -262862.67272, -77975.50382501, 317252]. This weight vector is equal to the weighted sum of all of the weight vectors from b, weighted by their counts. As such this weight vector is somewhat close to 5000 times the final weight vector in b because there were 500 test examples for 10 epochs. The average test error for the average Perceptron was also 0.014.

(d) [14 points] Compare the average prediction errors for the three methods. What do you conclude?

Both the voted and average Perceptron performed better than the standard Perceptron. This is because the data is not linearly separable and both the voted and averaged Perceptron use averaging to minimize the number of mistakes. This is effectively maximizing the soft margin of the classifier. Both the voted and average Perceptron perform the same because the learned weight vector at prediction time is exactly the same. At prediction time the voted Perceptron takes the sign of the weighted sum of the predictions of all of the learned weight vectors, weighted by the number of counts. The average Perceptron increments the learned weight vector with itself when the prediction is correct, effectively increasing the count for that weight vector. Therefore at prediction time the sign of the output of the learned weight vector of the average Perceptron will be the same the sign of the weighted sum of learned weight vectors of the voted Perceptron.