CS 5350/6350: Machine Learning Fall 2021

Homework 3

Handed out: 19 Oct, 2021
Due date: 11:59pm, 2 Nov, 2021

- 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).

   $dist(\mathbf{x}_i, h) = \frac{|\mathbf{w}^\top \mathbf{x}_i + b|}{||\mathbf{w}||}$

   (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?)

   Answer

   Figure 1 demonstrates that this hyperplane does separate the data. Table 1 (updated with Distance calculations) gives the distance between each data point and the hyperplane. The minimum distance is $\frac{1}{\sqrt{13}}$, which means that is the margin.

x_1	x_2	label	distance
1	1	1	$\frac{1}{\sqrt{13}}\|2*1+3*1-4\| = \frac{1}{\sqrt{13}}$
1	-1	-1	$\frac{1}{\sqrt{13}}\|2*1-3*1-4\| = \frac{5}{\sqrt{13}}$
0	0	-1	$\frac{1}{\sqrt{13}}\|2*0+3*0-4\| = \frac{4}{\sqrt{13}}$
-1	3	1	$\frac{1}{\sqrt{13}}\|2*-1+3*3-4\| = \frac{3}{\sqrt{13}}$

Table 1: Dataset 1, with distance calculations

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?

Answer

Figure 2 plots the hyperplane and the data points. Clearly, the hyperplane does not separate the data correctly, so there can be no margin.

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!

x_1	x_2	label	distance
-1	0	-1	$\frac{1}{\sqrt{2}}\|-1+0\| = \frac{1}{\sqrt{2}}$
0	-1	-1	$\frac{1}{\sqrt{2}}\|0-1\| = \frac{1}{\sqrt{2}}$
1	0	1	$\frac{1}{\sqrt{2}}\|1+0\| = \frac{1}{\sqrt{2}}$
0	1	1	$\frac{1}{\sqrt{2}}\|0+1\| = \frac{1}{\sqrt{2}}$

Table 3: Dataset 3, with distance calculations

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

Answer

Figure 3 plots Dataset 3 and a potential hyperplane that would separate the data. We can calculate R, which is the furthest point from the origin, but all points are equidistant from the origin, so they all have the same R. We see that $R = \sqrt{0^2 + 1^2} = 1$. If we use a simple hyperplane that goes through the origin and has the same distance to all four points, say $x_1 + x_2 = 0$, we see that $||\mathbf{w}|| = \sqrt{1^2 + 1^2} = \sqrt{2}$

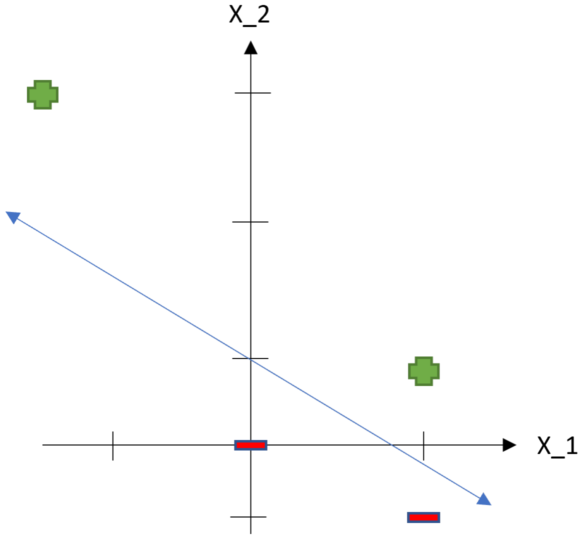

Figure 1: Weight vector change per update.

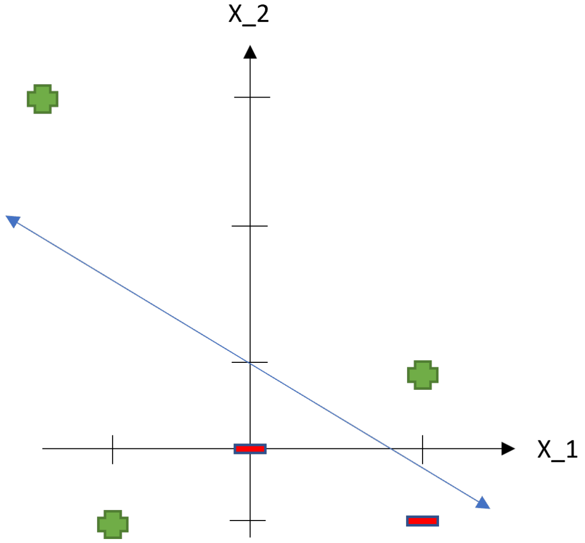

Figure 2: Weight vector change per update.

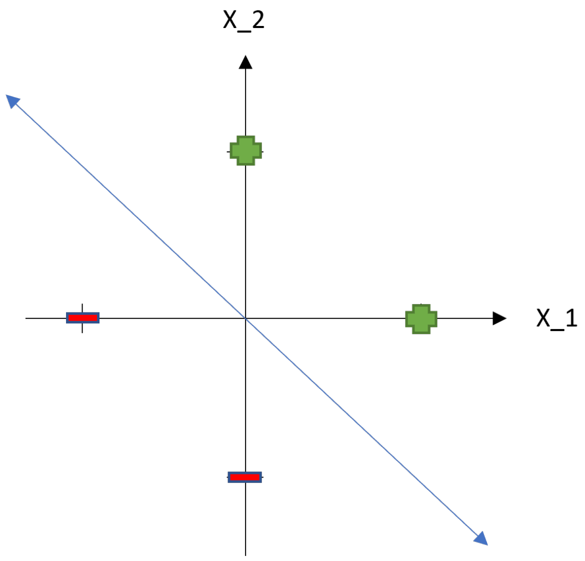

Figure 3: Weight vector change per update.

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.

Answer

Figure 4 plots Dataset 4. Clearly, the data is not linearly separable, so we cannot calculate any margin.

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.

Answer

Comparing to the proof in the lecture slides (CLICK HERE), the only difference is that $\mathbf{u}$ is not *necessarily* a unit vector. We will note that normally, the upper bound is $\frac{R^2}{\gamma^2}$.

We will pick up at part 3/3 of the proof seen in the lecture slides. We know that $\mathbf{u}^\top \mathbf{w}_t \geq t\gamma$ and $tR^2 \geq ||\mathbf{w}_t||^2$, the latter of which can be rewritten as $\sqrt{t}R \geq ||\mathbf{w}_t||$. We can also rewrite the dot product $\mathbf{u}^\top \mathbf{w}$ as $||\mathbf{u}||||\mathbf{w}_t||cos\theta$, where θ is the angle between

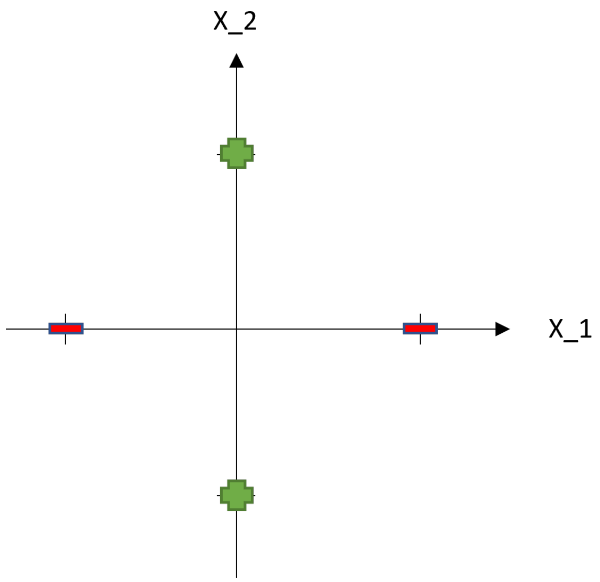

Figure 4: Weight vector change per update.

$\mathbf{u}$ and $\mathbf{w}_t$, and $cos\theta < 1$. The interesting case for us is when $||\mathbf{u}||||\mathbf{w}_t||cos\theta > ||\mathbf{w}_t||$, which means that $||\mathbf{u}|| > \frac{1}{cos\theta}$.

Obviously, based on Proof 3/3 from the lecture slides, if $||\mathbf{u}|| \leq \frac{1}{cos\theta}$, then we get $\frac{R^2}{\gamma^2}$.

If $||\mathbf{u}|| > \frac{1}{cos\theta}$,

$$=> ||\mathbf{u}||||\mathbf{w}_t||cos\theta \geq t\gamma$$

$$=> ||\mathbf{w}_t|| \geq \frac{t\gamma}{||\mathbf{u}||cos\theta}$$

From here, we add in that $\sqrt{t}R \geq ||\mathbf{w}_t||$ to yield,

$$R\sqrt{t} \geq ||\mathbf{w}_t|| \geq \frac{t\gamma}{||\mathbf{u}||cos\theta}$$

$$=> R^2 t \geq \frac{t^2\gamma^2}{||\mathbf{u}||^2 cos^2\theta}$$

$$=> ||\mathbf{u}||^2 cos^2\theta \frac{R^2}{\gamma^2} \geq t$$

So, it is the same value as the upper bound for when $\mathbf{u}$ is a unit vector, but scaled by the square of the norm of $\mathbf{u}$ and the cosine of its angle with $\mathbf{w}_t$. Obviously, this means it is simpler to compute the upper bound if we just keep $\mathbf{u}$ as a unit vector.

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

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

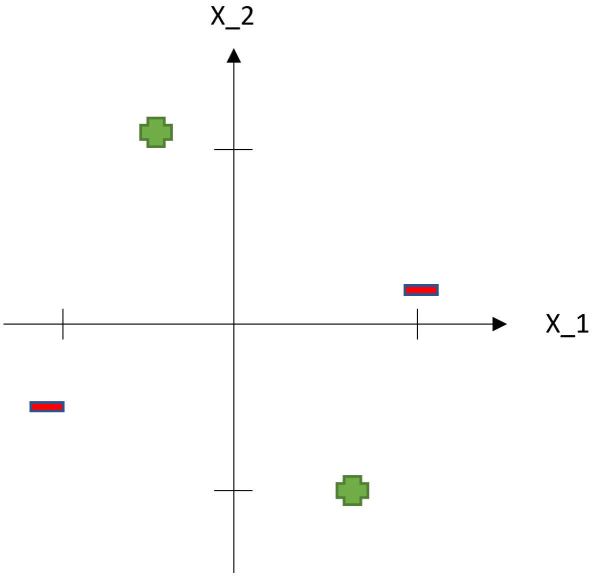

Figure 5: Weight vector change per update.

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.

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

   Answer

   Figure 5 shows a case with four points that cannot be shattered by any linear classifier.

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($\mathcal{H}$)?

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.

   Click here for the repository for HW3. It is not the same as the repository for HW1 or HW2.

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.

Answer

WEIGHT VECTOR

WaveletVariance -5.777450

WaveletSkew -3.204904

WaveletCurtosis -4.753204

ImageEntropy -0.603608

MODEL_BIAS -5.300000

ERRORS

TRAIN ERROR: 0.04243

TEST ERROR: 0.05000

(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.

Answer

Figure 6 shows how the weight vector changes with each update, and Figure 7 shows the votes assigned to each weight vector. The votes correspond to how many data points that each weight vector managed to accurately predict.

TRAIN ERROR: 0.01261

TEST ERROR: 0.01400

(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.

Answer

WEIGHT VECTOR WaveletVariance -37108.434439

WaveletSkew -25185.487162

WaveletCurtosis -25383.376175

ImageEntropy -7638.087261

MODEL_BIAS -34243.400000

ERRORS

TRAIN ERROR: 0.01491

TEST ERROR: 0.01400

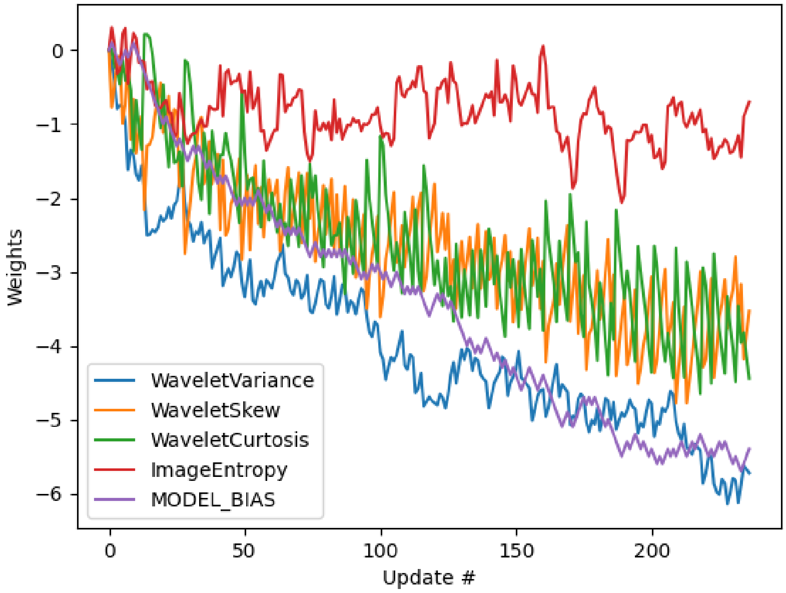

Figure 6: Weight vector change per update.

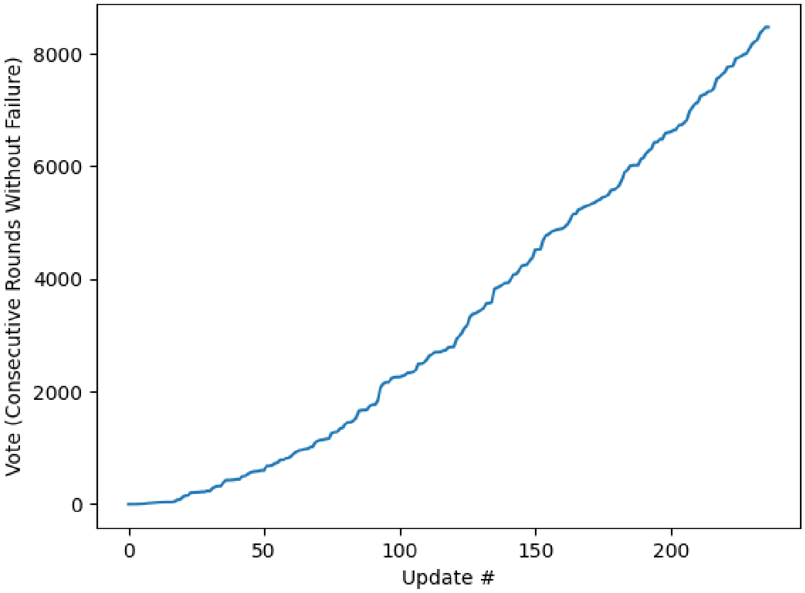

Figure 7: Votes for each weight vector.

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

Obviously, the standard Perceptron is the worst performer. But, Voted and Average Perceptrons perform about the same. Because the Voted Perceptron stores more data, the Average Perceptron is likely a more performative model at higher epochs since it only needs to compare to evaluate with one weight vector.