1. Perceptron Mistakes

Bookmark this page

Homework due Sep 29, 2020 19:59 EDT *Completed*

In this problem, we will investigate the perceptron algorithm with different iteration ordering.

Consider applying the perceptron algorithm **through the origin** based on a small training set containing three points:

$x^{(1)}$ =[-1,-1],	$y^{(1)}$=1
$x^{(2)}$ =[1,0],	$y^{(2)}$=-1
$x^{(3)}$ =[-1, 1.5],	$y^{(3)}$=1

Given that the algorithm starts with $\theta^{(0)} = 0$, the first point that the algorithm sees is always considered a mistake. The algorithm starts with some data point and then cycles through the data (in order) until it makes no further mistakes.

1. (a)

4.0/4 points (graded)

How many mistakes does the algorithm make until convergence if the algorithm starts with data point $x^{(1)}$? How many mistakes does the algorithm make if it starts with data point $x^{(2)}$?

Also provide the progression of the separating plane as the algorithm cycles in the following **list format**: $[[\theta_1^{(1)},\theta_2^{(1)}],\ldots,[\theta_1^{(N)},\theta_2^{(N)}]]$, where the superscript denotes different θ as the separating plane progresses. For example, if θ progress from $[0,0]$ (initialization) to $[1,2]$ to $[3,-2]$, you should enter $[[1,2],[3,-2]]$

Please enter the **number of mistakes** of Perceptron algorithm if the algorithm starts with $x^{(1)}$.

2 ✓

Please enter the **progression of the separating hyperplane (θ, in the list format described above)** of Perceptron algorithm if the algorithm starts with $x^{(1)}$.

()

[[-1,-1],[-2,0.5]] ✓

Please enter the **number of mistakes** of Perceptron algorithm if the algorithm starts with $x^{(2)}$.

1 ✓

Please enter the **progression of the separating hyperplane (θ, in the list format described above)** of Perceptron algorithm if the algorithm starts with $x^{(2)}$.

[[-1,0]] ✓

Solution:

- If the algorithm starts with $x^{(1)}$, then it will encouter two errors: $x^{(1)}$ and $x^{(3)}$.
- Since $[0,0] \cdot x^{(1)} = 0$, the result is not greater than 0.
- It would induce an update $\theta^{(1)} = \theta^{(0)} + y^{(1)}x^{(1)} = [-1,-1]$.
- Then $\theta^{(1)} \cdot x^{(2)} < 0$, so there is no mistakes.
- Finally, $\theta^{(1)} \cdot x^{(3)} < 0$, which incur an error and would induce an update $\theta^{(2)} = \theta^{(1)} + y^{(3)}x^{(3)}$.
- Afterwards, there should be no mistakes.
- If the algorithm starts with $x^{(2)}$, then it will only encounter one error: $x^{(2)}$. The derivation is similar to the above analysis.
- The progression of the separating hyperplane reflects the updated hyperplane after encountering errors.

Show answer

Submit You have used 1 of 3 attempts

Answers are displayed within the problem

1. (b)

1/1 point (graded)

In part (a), what are the factors that affect the number of mistakes made by the algorithm?

Note: Only choose factors that were changed in part (a), **not** all factors that can affect the number of mistakes

(Choose all that apply.)

- [✓ Iteration order]
- Maximum margin between positive and negative data points
- Maximum norm of data points

✓

Solution:

- Only the iteration order is changed in part (a).

Show answer

Submit You have used 1 of 3 attempts

1. (c)

4.0/4 points (graded)

Now assume that $x^{(3)} = [-1, 10]$. How many mistakes does the algorithm make until convergence if cycling starts with data point $x^{(1)}$?

Also provide the progression of the separating plane as the algorithm cycles in the following **list format**: $[[\theta_1^{(1)}, \theta_2^{(1)}], \ldots, [\theta_1^{(N)}, \theta_2^{(N)}]]$, where the superscript denotes different θ as the separating plane progresses. For example, if θ progress from $[0, 0]$ (initialization) to $[1, 2]$ to $[3, -2]$, you should enter $[[1, 2], [3, -2]]$

Please enter the **number of mistakes** of Perceptron algorithm if the algorithm starts with $x^{(1)}$.

6 ✓

Please enter the **progression of the separating hyperplane (θ, in a list format described above)** of Perceptron algorithm if the algorithm starts with $x^{(1)}$.

[[-1,-1],[-2,9],[-3,8],[-4,7],[-5, ✓

Please enter the **number of mistakes** of Perceptron algorithm if the algorithm starts with $x^{(2)}$.

1 ✓

Please enter the **progression of the separating hyperplane (θ, in the list format described above)** of Perceptron algorithm

1. (d)

1/1 point (graded)

For a fixed iteration order, what are the factors that affect the number of mistakes made by the algorithm between part (a) and part (c)?

Note: Only choose factors that were changed between part (a) and part (c), **not** all factors that can affect the number of mistakes

(Choose all that apply.)

☐ Iteration order

☐ Maximum margin between positive and negative data points

[✓ Maximum norm of data points]

✓

Solution:

- The maximum norm of data points cause part (c) to make more mistakes.

Show answer

1. (e) (Optional)

0 points possible (ungraded)

In 1962, Novikoff has proven the following theorem.

Assume:

- There exists θ^* such that $\frac{y^{(i)}(\theta^* x^{(i)})}{\|\theta^*\|} \geq \gamma$ for all $i = 1, \cdots, n$ and some $\gamma > 0$
- All the examples are bounded $\|x^{(i)}\| \leq R, i = 1, \cdots, n$

Then the number k of updates made by the perceptron algorithm is bounded by $\frac{R^2}{\gamma^2}$.

(Note that the first condition implies that the data is linearly separable)

For proof, refer to theorem 1 of this paper. Based on this theorem, what are the factors that constitute the bound on the number of mistakes made by the algorithm?

(Choose all that apply.)

☐ Iteration order

☑ Maximum margin between positive and negative data points

☑ Maximum norm of data points

☐ Average norm of data points

✓

Solution:

- Iteration order would affect relative convergence speed, but does not constitute the bound in this theorem.
- The maximum margin and the maximum norm of data points consitute the bounds in the maximum number of mistakes.
- We can always scale an easy dataset to achieve the same average norm of data points with the same number of mistakes.

Show answer

Submit You have used 2 of 3 attempts

Answers are displayed within the problem

1. (f) (Optional)

0 points possible (ungraded)

1. (f) (Optional)

0 points possible (ungraded)

Now we want to establish an adversarial procedure to maximize the number of mistakes the perceptron algorithm makes. What are possible solutions? You should consider a general dataset instead of part (a) and part (c). (Choose all that apply.)

- [✓] Exhaustive search the worst ordering
- [✓] Dynamic Programming the worst ordering
- [ ] Greedily select the data point with the maximum norm

Solution:

- There are only finitely possible iteration orders, since the algorithm can converge in finite adjustments. As a direct result, exhaustive search can always find the worst ordering.
- Given any prior mistakes made by the algorithm, we know that the maximum number of mistakes should be prior mistakes plus maximum future mistakes. Hence, optimal substructure exist and dynamic programming can be applied.
- Choosing the data point with maximum norm does not maximimze the number of mistakes. For instance, take example from part (c) and adjust $x^{(2)}$ to be $[100000, 0]$ to get a counter example.

2. Perceptron Performance

Bookmark this page

Homework due Sep 29, 2020 19:59 EDT Completed

In class we initialized the perceptron algorithm with $\theta = 0$. In this problem we will also explore other initialization choices.

---

2. (a)

2.0/2 points (graded)

The following table shows a data set and the number of times each point is misclassified during a run of the perceptron algorithm (**with offset** θ_0). θ and θ_0 are initialized to zero.

i	$x^{(i)}$	$y^{(i)}$	times misclassified
1	[-4, 2]	+1	1
2	[-2, 1]	+1	0
3	[-1, -1]	-1	2
4	[2, 2]	-1	1
5	[1, -2]	-1	0

Write down the state of θ and θ_0 after this run has completed (note, the algorithm may not yet have converged). Enter θ as a list $[\theta_1, \theta_2]$ and θ_0 as a single number in the following boxes.

Please enter θ:

[-4,2] ✓ **Answer:** [-4, 2]

Please enter θ_0 :

-2 ✓ **Answer:** -2

Solution:

- Since perceptron update rule updates θ simply by adding $x^{(i)}y^{(i)}$, the resulting θ should be the summation of all mistakes.
- Additional Insight: since perceptron update rule is additively associative, doing updates in any order would lead to the same result.

Show answer

Submit You have used 1 of 3 attempts

Answers are displayed within the problem

2. (b)

2/2 points (graded)

Provide one example of a different initialization of θ such that the perceptron algorithm with this initialization would not produce any mistakes during a run through the data.

$[\theta_1, \theta_2]$: [-4,2] ✓ **Answer:** See solution

θ_0: -3 ✓ **Answer:** See solution

Solution:

The answer (θ, θ_0) should be such that:

- $-4\theta_1 + 2\theta_2 + \theta_0 > 0$
- $-2\theta_1 + \theta_2 + \theta_0 > 0$
- $-\theta_1 - \theta_2 + \theta_0 < 0$
- $2\theta_1 + 2\theta_2 + \theta_0 < 0$
- $1\theta_1 - 2\theta_2 + \theta_0 < 0$

For instance, any line strictly inside the boundry $x_2 = x_1$ and $x_2 = x_1 + 3$ are valid solutions.

2. (c)

2.0/3 points (graded)

The theorem from question 1. (e) provides an upper bound on the number of steps of the Perceptron algorithm and implies that it indeed converges. In this question, we will show that the result still holds even when θ is not initialized to 0.

In other words: Given a set of training examples that are linearly separable through the origin, show that the initialization of θ does not impact the perceptron algorithm's ability to eventually converge.

To derive the bounds for convergence, we assume the following inequalities holds:

- There exists θ^* such that $\frac{y^{(i)}\,(\theta^* x^{(i)})}{\|\theta^*\|} \geq \gamma$ for all $i = 1, \cdots, n$ and some $\gamma > 0$
- All the examples are bounded $\|x^{(i)}\| \leq R, i = 1, \cdots, n$

If θ is initialized to 0, we can show by induction that:

$$\theta^{(k)} \cdot \frac{\theta^*}{\|\theta^*\|} \geq k\gamma$$

For instance,

$$\theta^{(k+1)} \cdot \frac{\theta^*}{\|\theta^*\|} = (\theta^{(k)} + y^{(i)} x^{(i)}) \cdot \frac{\theta^*}{\|\theta^*\|} \geq (k+1)\,\gamma$$

If we initialize θ to a general (not necessarily 0) $\theta^{(0)}$, then:

$$\theta^{(k)} \cdot \frac{\theta^*}{||\theta^*||} \geq a + k\gamma$$

Determine the formulation of a in terms of θ^* and $\theta^{(0)}$:

Important: Please enter θ^* as `theta^{star}` and $\theta^{(0)}$ as `theta^{0}`, and use `norm(...)` for the vector norm $||\ldots||$.

$a =$ (theta^{0}*theta^{star})/norm(theta^{star}) ✓

Answer: theta^{0}*theta^{star} / norm(theta^{star})

$$\frac{\theta^0 \cdot \theta^\star}{||\theta^\star||}$$

If θ is initialized to 0, we can show by induction that:

$$||\theta^{(k)}||^2 \leq kR^2$$

For instance,

$$||\theta^{(k+1)}||^2 \leq ||\theta^{(k)} + y^{(i)}x^{(i)}||^2 \leq ||\theta^{(k)}||^2 + R^2$$

If we initialize θ to a general (not necessarily 0) $\theta^{(0)}$, then:

$$||\theta^{(k)}||^2 \leq kR^2 + c^2$$

Determine the formulation of c^2 in terms of $\theta^{(0)}$:

$c^2 =$ norm(theta^{0})^2 ✓ **Answer:** norm(theta^{0})^2

$||\theta^0||^2$

From the above inequality, we can derive the inequality $||\theta^{(k)}|| \leq c + \sqrt{k}R$ by applying the following inequality: $\sqrt{x^2+y^2} \leq \sqrt{(x+y)^2}$ if $x, y > 0$.

If θ is initialized to 0, we then use the fact that $1 \geq \frac{\theta^{(k)}}{||\theta^{(k)}||} \cdot \frac{\theta^*}{||\theta^*||}$ to get the upper bound $k \leq \frac{R^2}{\gamma^2}$.

In the case where we initialize θ to a general $\theta^{(0)}$, use the inequality for $\theta^{(k)} \cdot \frac{\theta^*}{||\theta^*||}$ above and the inequality $||\theta^{(k)}|| \leq c + \sqrt{k}R$ to derive a bound on the number of iterations k.

Hint: Use the larger root of a quadratic equation to obtain the upper bound.

Note: Give your answer in terms of a, c, R, γ (enter the latter as `gamma`).

$k \leq$

(R*sqrt(R^2+4*(c-a)*gamma) + R^2 + 2*(c-a)*gamma) / (2*gamma^2)

Answer: (R*sqrt(R^2+4*(c-a)*gamma) + R^2 + 2*(c-a)*gamma) / (2*gamma^2)

$$\frac{R\cdot\sqrt{R^2+4\cdot(c-a)\cdot\gamma}+R^2+2\cdot(c-a)\cdot\gamma}{2\cdot\gamma^2}$$

STANDARD NOTATION

Solution:

The first bound follows by recursion of $\theta^k \cdot \frac{\theta^*}{\|\theta^*\|} \geq \theta^{k-1} \cdot \frac{\theta^*}{\|\theta^*\|} + \gamma$.

The second bound follows by recursion of $\|\theta^k\|^2 \leq \|\theta^{k-1}\|^2 + R^2$.

The final bound is obtained by solving the inequality $1 \geq \theta^k \cdot \frac{\theta^*}{\|\theta^k\|\|\theta^*\|} \geq \frac{a+k\gamma}{c+\sqrt{k}R}$, i.e. $a + k\gamma - c \leq \sqrt{k}R$.

At this point, you can square both sides and solve the quadratic equation to get the upper bound.

Alternatively, solve the quadratic equation for $\sqrt{k}$ and square the answer to get the desired upper bound:

$$k \leq \frac{\left(R + \sqrt{R^2 - 4\gamma(a-c)}\right)^2}{4\gamma^2}$$

Show answer

Submit You have used 1 of 3 attempts

2. (d)

2/2 points (graded)

Since the convergence of the perceptron algorithm doesn't depend on the initialization, the end performance on the training set must be the same. Are the resulting θ's the same regardless of the initialization?

Yes [✓ No]

✓

Does this necessarily imply that the performance on a test set is the same?

Yes [✓ No]

✓

Solution:

- Any distinct θ that can separate the data are valid solutions, so there are infinitely many different valid correct θ in general given that the data can be separated by more than 1 line.

- Two different θ would always make different predictions for a testing data point between the two lines, so the testing performance is always different for a testing dataset that contains exactly this point.

Show answer

Submit You have used 1 of 3 attempts

Answers are displayed within the problem

3. Decision Boundaries

Bookmark this page

Homework due Sep 29, 2020 19:59 EDT Completed

In this problem, we will investigate the decision boundary of different classifiers.

3. (a)

2/2 points (graded)

Consider the function defined over three binary variables: $f(x_1, x_2, x_3) = (\neg x_1 \wedge \neg x_2 \wedge \neg x_3)$.

We aim to find a θ such that, for any $x = [x_1, x_2, x_3]$, where $x_i \in \{0, 1\}$:

$$\theta \cdot x + \theta_0 > 0 \text{ when } f(x_1, x_2, x_3) = 1, \text{ and}$$

$$\theta \cdot x + \theta_0 < 0 \text{ when } f(x_1, x_2, x_3) = 0.$$

If $\theta_0 = 0$ (no offset), would it be possible to learn such a θ?

◯ Yes

[✓ No]

✓

Would it be possible to learn the pair θ and θ_0?

[✓ Yes]

No

✓

Solution:

- Since $\theta \cdot 0 = 0$, it is impossible to obtain $\theta \cdot x + \theta_0 > 0$ for $f(0,0,0) = 1$.
- $\theta_1 = \theta_2 = \theta_3 = -1$ and $\theta_0 = 0.5$ is a valid solution.

Show answer

Submit You have used 2 of 3 attempts

Answers are displayed within the problem

3. (b-1)

1/1 point (graded)

You are given the following labeled data points:

- Positive examples: $[-1, 1]$ and $[1, -1]$,
- Negative examples: $[1, 1]$ and $[2, 2]$.

For each of the following parameterized families of classifiers, identify which parameterized family has a family member that can correctly classify the above data and find the corresponding parameters of a family member that can correctly classify the above data.

Note: If there is no family member inside the parameterized family that can correctly classify the above data, just enter 0 for all the parameters.

Inside (positive) or outside (negative) of an origin-centered circle with radius r. Enter a scalar for r. If there is no such r, just enter 0.

0 ✓ **Answer:** 0

Solution:

- Any circle that correctly classifies $[-1, 1]$ and $[1, -1]$ would incorrectly classify $[1, 1]$

Show answer

3. (b-2)

2/2 points (graded)

Inside (positive) or outside (negative) of an $[x, y]$-centered circle with radius r.

$[x, y]$: [-1,-1] ✓ **Answer:** See solution

r: 2*sqrt(1.5) ✓ **Answer:** See solution

Solution:

- A valid solution is $[x, y] = [-1, -1], r = 2.1$

Show answer

Submit You have used 1 of 3 attempts

Answers are displayed within the problem

3. (b-3)

3. (b-3)

1.0/1 point (graded)

Strictly above (positive) or below (negative) a line through the origin with normal θ. Here we define "above" as $\theta \cdot x > 0$, and define "below" similarly. **Note:** Please enter a list for θ as $[\theta_1, \theta_2]$. If there is no solution, enter $[0, 0]$

[0,0] ✓ **Answer:** [0, 0]

Solution:

- There is no line through the origin that can simultaneously be strictly below $[1, -1]$ and $[-1, 1]$

Show answer

Submit You have used 1 of 3 attempts

Answers are displayed within the problem

3. (b-4)

0/2 points (graded)

3. (b-4)

0/2 points (graded)

Strictly above (positive) or below (negative) a line with normal θ and offset θ_0. Here we define "above" as $\theta \cdot x + \theta_0 > 0$, and define "below" similarly. **Note:** If there is no solution, enter $\theta = [0, 0]$ and $\theta_0 = 0$.

$[\theta_1, \theta_2]$: [-1,=1] **Answer:** See solution

θ_0: 0.5 **Answer:** See solution

Solution:

- A valid solution is $[\theta_1, \theta_2, \theta_0] = [-1, -1, 0.5]$

Show answer

Submit You have used 2 of 3 attempts

3. (b-5)

1/1 point (graded)

Which of the below are families of linear classifiers?

(Choose all that apply.)

- ☐ Inside or outside of an origin-centered circle with radius r.
- ☐ Inside or outside of an $[x, y]$-centered circle with radius r.
- ☑ Strictly above or below a line through the origin with normal θ.
- ☑ Strictly above or below a line with normal θ and offset θ_0.

✓

Solution:

- The first two families are nonlinear (circles), and the last two families are linear classifiers (lines).

Show answer

4. Linear Support Vector Machines

Bookmark this page

Homework due Sep 29, 2020 19:59 EDT Completed

In this problem, we will investigate minimizing the training objective for a Support Vector Machine (with margin loss).

The training objective for the Support Vector Machine (with margin loss) can be seen as optimizing a balance between the average hinge loss over the examples and a regularization term that tries to keep the parameters small (increase the margin). This balance is set by the regularization parameter $\lambda > 0$. Here we only consider the case without the offset parameter θ_0 (setting it to zero) so that the training objective is given by

$$\left[\frac{1}{n}\sum_{i=1}^{n} Loss_h\,(y^{(i)}\,\theta\cdot x^{(i)}\,)\right] + \frac{\lambda}{2}\|\theta\|^2 = \frac{1}{n}\sum_{i=1}^{n}\left[Loss_h\,(y^{(i)}\,\theta\cdot x^{(i)}\,) + \frac{\lambda}{2}\|\theta\|^2\right] \tag{4.3}$$

where the hinge loss is given by

$$\text{Loss}_h\,(y\,(\theta\cdot x)) = \max\{0, 1 - y\,(\theta\cdot x)\}$$

$$\hat{\theta} = \text{Argmin}_\theta\,[\text{Loss}_h\,(y\,\theta\cdot x\,) + \frac{\lambda}{2}\|\theta\|^2] \tag{4.4}$$

Note: For all of the exercises on this page, assume that $n = 1$ where n is the number of training examples and $x = x^{(1)}$ and

Minimizing Loss - Case 1

1/1 point (graded)

In this question, suppose that $\text{Loss}_h\left(y\left(\hat{\theta}\cdot x\right)\right) > 0$. Under this hypothesis, solve for optimisation problem and express $\hat{\theta}$ in terms of x, y and λ

(x*y)/lambda

✓ **Answer:** x*y/lambda

$\frac{x\cdot y}{\lambda}$

STANDARD NOTATION

Solution:

$$\hat{\theta} = \text{Argmin}_\theta\left[\text{Loss}_h\left(y\,\theta\cdot x\,\right) + \frac{\lambda}{2}||\theta||^2\right]$$

The above loss can be minimized by solving for the following equation

$$0 = \nabla_\theta\left[\text{Loss}_h\left(y\left(\theta\cdot x\right)\right)\right] + \nabla_\theta\left[\frac{\lambda}{2}||\theta||^2\right]$$

Given that

$$\begin{aligned}
\text{Loss}_h\left(y\left(\hat{\theta}\cdot x\right)\right) &> 0 \\
\text{Loss}_h\left(y\left(\hat{\theta}\cdot x\right)\right) &= \max\{0, 1-y\left(\theta\cdot x\right)\} \\
\text{Loss}_h\left(y\left(\hat{\theta}\cdot x\right)\right) &= 1-y\left(\theta\cdot x\right) \\
\nabla_\theta\left[\text{Loss}_h\left(y\left(\theta\cdot x\right)\right)\right] &= -yx
\end{aligned}$$

Plugging this back in the previous equation, we get:

$$0 = \lambda\hat{\theta} - yx$$

$$\hat{\theta} = \frac{1}{\lambda}yx$$

Show answer

Submit You have used 1 of 3 attempts

Answers are displayed within the problem

Minimizing Loss - Numerical Example (1)

Consider minimizing the above objective fuction for the following numerical example:

$$\lambda = 0.5, y = 1, x = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Note that this is a classification problem where points lie on a two dimensional space. Hence $\hat{\theta}$ would be a two dimensional vector.

Let $\hat{\theta} = \left[\hat{\theta_1}, \hat{\theta_2}\right]$, where $\hat{\theta_1}, \hat{\theta_2}$ are the first and second components of $\hat{\theta}$ respectively.

Solve for $\hat{\theta_1}, \hat{\theta_2}$.

Hint: For the above example, show that $\text{Loss}_h\left(y\left(\hat{\theta} \cdot x\right)\right) \leq 0$

$\hat{\theta_1} =$

1 ✓ **Answer:** 1.0

$\hat{\theta_2} =$

0 ✓ **Answer:** 0.0

Solution:

First note that for this example $Loss_h\left(y\left(\theta\cdot x\right)\right)\leq 0$.
To show this we use proof by contradiction.

Suppose $Loss_h\left(y\left(\theta\cdot x\right)\right)>0$:
From the previous problem, we know that under this condition, $\hat{\theta}=\frac{yx}{\lambda}$
For this example, $\hat{\theta}=\begin{bmatrix}2\\0\end{bmatrix}$.
For this value of $\hat{\theta}$, we see that $1-\left(y\left(\theta\cdot x\right)\right)=1-2=-1<0$ contradicting our original assumption.

Hence, $\mathrm{Loss}_h\left(y\left(\theta\cdot x\right)\right)\leq 0$, which implies that $y\left(\theta\cdot x\right)\geq 1$.

We are left with minimizing $\frac{\lambda}{2}||\theta||^2$ under the constraint $y\left(\theta\cdot x\right)\geq 1$.

The geometry of the problem implies that in fact, $y\left(\theta\cdot x\right)=1$.

That is, $1-\left(\hat{\theta_1}*1+\hat{\theta_2}*0\right)=0$ implying that $\hat{\theta_1}=1$.

Then, to minize $||\theta||$, $\hat{\theta_2}=0$.

Therefore $\hat{\theta}=\begin{bmatrix}1\\0\end{bmatrix}$.

In fact, we can show that $th\hat{e}ta=\frac{x}{y||(||x)^2}$. Looking back at the previous question, the solution of the optimization is then necessarily of the form $\hat{\theta}=\eta yx$ for some real $\eta>0$.

Minimizing Loss - Numerical Example (2)

0/1 point (graded)

Now, let $\hat{\theta} = \hat{\theta}(\lambda)$ be the solution as a function of λ.

For what value of $\|x\|^2$, the training example (x, y) will be misclassified by $\hat{\theta}(\lambda)$?

$\|x\|^2 =$ 0 **Answer:** 0

0

Solution:

For a point to be considered misclassified

$$y\hat{\theta} \cdot x \leq 0$$

The above condition implies that the hinge loss is greater than zero. From above problems, we know that under this condition,

$$\hat{\theta} = \frac{yx}{\lambda}$$

$$y\hat{\theta} \cdot x = \frac{y^2\|x\|^2}{\lambda} \leq 0$$

All terms of the product are non-negative, making it impossible to be < 0. But if $\|x\| = 0$, the product can be 0.

Hence $\|x\|^2 = 0$

Show answer

Submit You have used 1 of 3 attempts

5. Passive-aggressive algorithm

Homework due Sep 29, 2020 19:59 EDT *Completed*

In this problem, we will try to understand the loss in Passive-Aggressive (PA) Perceptron algorithm.

The passive-aggressive (PA) algorithm (without offset) responds to a labeled training example (x, y) by finding θ that minimizes

$$\frac{\lambda}{2}\left\|\theta-\theta^{(k)}\right\|^2+\text{Loss}_h\left(y\theta\cdot x\right) \tag{4.5}$$

where $\theta^{(k)}$ is the current setting of the parameters prior to encountering (x, y) and

$$\text{Loss}_h\left(y\theta\cdot x\right)=\max\{0,1-y\theta\cdot x\}$$

is the hinge loss. We could replace the loss function with something else (e.g., the zero-one loss). The form of the update is similar to the perceptron algorithm, i.e.,

$$\theta^{(k+1)}=\theta^{(k)}+\eta\, yx \tag{4.6}$$

but the real-valued step-size parameter η is no longer equal to one; it now depends on both $\theta^{(k)}$ and the training example

(x, y).

Update equation

1/1 point (graded)

Consider minimizing the above defined loss function with the hinge loss component.
What happens to the step size at large values of λ? Please choose one from the options below:

- If λ is large, the step-size of the algorithm (η) would be large
- [✓ If λ is large, the step-size of the algorithm (η) would be small]

✓

Solution:

As λ increases, the passive-aggressive algorithm is dissuaded from updating very far from the previous θ. This is because λ serves as the weight for the regularization term (the portion that prevents overfitting: $|| \ \theta - \theta^{(k)} \ ||$) of the minimizing function. Thus, as λ increases, we expect the step size between updates to decrease, so η should be smaller.

Show answer

Submit You have used 2 of 3 attempts

Calculating the Step size

1/1 point (graded)

Suppose $\text{Loss}_h\left(y\theta^{(k+1)} \cdot x\right) > 0$ after the update. Express the value of η in terms of λ in this case. (Hint: you can simplify the loss function in this case).

1/lambda

✓ **Answer:** 1/lambda

$\frac{1}{\lambda}$

STANDARD NOTATION

Solution:

When $\text{Loss}_h\left(y\theta^{(k+1)} \cdot x\right) > 0$, the hinge loss,

$$\max\{0, 1 - y\theta \cdot x\}$$

can be simply rewritten as $1 - y\theta \cdot x$. Thus, our minimizing function simplifies as follows:

$$\begin{aligned} f(\theta) &= \frac{\lambda}{2}\left\|\theta - \theta^{(k)}\right\|^2 + \text{Loss}_h\left(y\theta \cdot x\right) \\ &= \frac{\lambda}{2}\left\|\theta - \theta^{(k)}\right\|^2 + 1 - y\theta \cdot x \end{aligned}$$

We compute the minimum by setting gradient of $f(\theta)$ w.r.t. θ to 0:

$$\begin{aligned}\nabla_\theta f &= \lambda(\theta - \theta^{(k)}) - yx = 0 \\ (\theta - \theta^{(k)}) &= \frac{1}{\lambda} yx \\ \theta &= \theta^{(k)} + \frac{1}{\lambda} yx\end{aligned}$$

where $\frac{1}{\lambda}$ is just the step size.
Thus, $\eta = \frac{1}{\lambda}$ when $\text{Loss}_h(y\theta^{(k+1)} \cdot x) > 0.$

Show answer

Submit You have used 1 of 3 attempts

Answers are displayed within the problem

Loss functions and decision boundaries

1.0/1.0 point (graded)

Consider minimizing the above defined loss function and the setting of our decision boundary plotted below. We ran our PA algorithm on the next data point in our sequence - a positively-labeled vector (indicated with a $+$). We plotted the results of our algorithm after the update, by trying out a few different variations of loss function and λ as follows:

1. hinge loss and a large λ

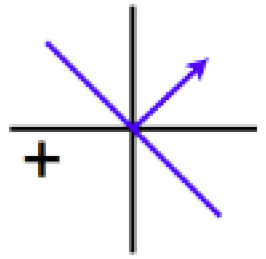
2. hinge loss and a small λ

3. 0-1 loss and a large λ

4. 0-1 loss and a small λ

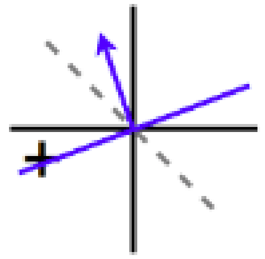
Each of the options below provides a matching between the 4 variations above with a decision boundary plotted in a-d below. Please choose the options that match them correctly. Note that the dotted lines correspond to the previous decision boundary, and the solid blue lines correspond to the new decision boundary; also, note that these are just sketches which ignore any changes to the magnitude of θ.

setting before update:

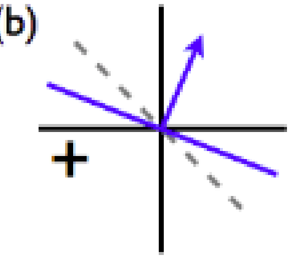


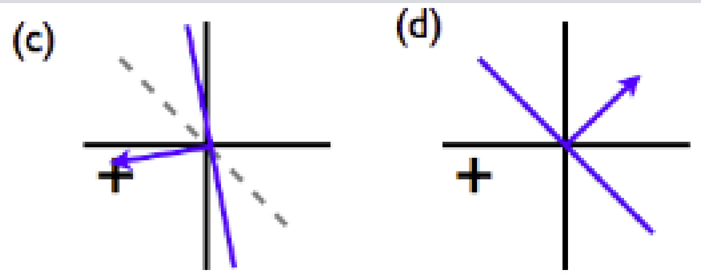
possible settings after update:

(a)



(b)





(a) 0-1 loss, small lambda ✓ **Answer:** 0-1 loss, small lambda

(b) Hinge loss, large lambda ✓ **Answer:** Hinge loss, large lambda

(c) Hinge loss, small lambda ✓ **Answer:** Hinge loss, small lambda

(d) 0-1 loss, large lambda ✓ **Answer:** 0-1 loss, large lambda

Solution:

- 1 - b hinge loss and a large λ. For hinge loss, loss can be improved by moving the θ vector towards the example. However, since λ is large, the change in θ term dominates, so θ only rotates slightly.

- 2 - c hinge loss and a small λ. For hinge loss, loss can be improved by moving the θ vector towards the example. Since λ is small, the loss term dominates, so the loss is minimized when the example is correctly classified. The difference between hinge loss and 0-1 loss with small λ is that hinge loss will correctly classify the example with a margin from the boundary. This is because hinge loss is minimized when $y\theta \cdot x \geq 1$ while 0-1 is when $y\theta \cdot x \geq 0$.

- 3 - d 0-1 loss and a large λ. Since the loss function is zero-one, it can only take on two values whether its classified correctly or not. With λ large, the change in θ term dominates. In order to minimize the change in θ, θ simply stays the same after the update and does not improve the loss at all.

- 4 - a 0-1 loss and a small λ. Since λ is small, the zero-one loss term dominates. Since the loss function can only improve when it is classified correctly, the example must be on the positive side. Because all correctly classified examples show the same loss, the secondary goal of minimizing the change in θ will cause the point to be immediately on the positive side of the decision boundary.

Show answer

Submit You have used 3 of 3 attempts

6. Perceptron Updates

Homework due Sep 29, 2020 19:59 EDT *Completed*

In this problem, we will try to understand the convergence of perceptron algorithm and its relation to the ordering of the training samples for the following simple example.

Consider a set of $n = d$ labeled $d-$dimensional feature vectors, $\{(x^{(t)}, y^{(t)}), t = 1, \ldots, d\}$ defined as follows:

$$x_i^{(t)} = \cos(\pi t) \quad \text{if} \quad i = t \tag{4.7}$$

$$x_i^{(t)} = 0 \quad \text{otherwise,} \tag{4.8}$$

Recall the no-offset perceptron algorithm, and assume that $\theta \cdot x = 0$ is treated as a mistake, regardless of label. Assume that in all of the following problems, we initialize $\theta = 0$ and when we refer to the perceptron algorithm we only consider the no-offset variant of it.

Working out Perceptron Algorithm

3/3 points (graded)

Consider the $d = 2$ case. Let $y^{(1)} = 1, y^{(2)} = 1$. Assume that the feature vector $x^{(1)}$ is presented to the perceptron algorithm before $x^{(2)}$.

For this particular assignment of labels, work out the perceptron algorithm until convergence.

Let $\hat{\theta}$ be the resulting θ value after convergence. Note that for $d = 2$, $\hat{\theta}$ would be a two-dimensional vector. Let's denote the first and second components of $\hat{\theta}$ by $\hat{\theta}_1$ and $\hat{\theta}_2$ respectively.

Please enter the total number of updates made to θ by perceptron algorithm:

2

✓ **Answer:** 2

Please enter the numerical value of $\hat{\theta}_1$:

-1

✓ **Answer:** -1

Please enter the numerical value of $\hat{\theta}_2$:

1

✓ **Answer:** 1

Solution:

The first iteration of perceptron with data point $(x^{(i)}, y^{(i)})$ will be a mistake due to our initialization of $\theta^{(0)} = \bar{0}$. The first update sets $\theta^{(1)} = y^{(i)}x^{(i)} = x^{(i)}$. The second iteration of perceptron with data point $(x^{(j)}, y^{(j)})$ will also yield a mistake since

$$\theta^{(1)} \cdot x^{(j)} = x^{(i)} \cdot x^{(j)} = 0$$

. Thus, for the $d = 2$ example, after 2 updates $\theta^{(2)} = x^{(i)} + x^{(j)}$. We now check whether the second pass yields mistakes

$$\begin{aligned} y^{(i)}\theta^{(2)} \cdot x^{(i)} &= y^{(i)}\left(y^{(i)}x^{(i)} + y^{(j)}x^{(j)}\right) \cdot x^{(i)} \\ &= \left\|x^{(i)}\right\|^2 + x^{(i)} \cdot x^{(j)} \\ &= \left\|x^{(i)}\right\|^2 + 0 \\ &> 0 \end{aligned}$$

so the first point is classified correctly. Similarly, the second point is also classified correctly.

Therefore, it only takes two updates to classify the $d = 2$ dataset and θ converges to $\hat{\theta} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$

Show answer

Submit You have used 2 of 3 attempts

Answers are displayed within the problem

General Case - Number of updates

Now consider any $d > 0$.

For the specific dataset we are considering, please choose the correct answer from the options below:

[✓ Perceptron algorithm will make exactly d updates to θ regardless of the order and labelings of the feature vectors]

Perceptron algorithm will make at least d updates to θ with the exact number of updates depending on the ordering of the feature vectors presented to it

Perceptron algorithm will make at least d updates to θ with the exact number of updates depending on the ordering of the feature vectors presented to it and their labeling

Perceptron algorithm will make at most d updates to θ with the exact number of updates depending on the ordering of the feature vectors presented to it and their labeling

✓

Solution:

Using the intuition that we gained from the $d = 2$ case, we notice that upon the first pass of the data points, every $\theta \cdot x^{(i)} = 0$ for all i. In other words, each data point we consider in sequence lies on the current boundary. Since each point starts as a mistake, there are $\geq d$ updates. Now it remains to be shown that after d updates, all points are classified correctly. After the ith update, we add $y^{(i)} x^{(i)}$ to $\theta^{(i-1)}$. After d updates,

$$\theta^{(d)} = \sum_{i=1}^{d} y^{(i)} x^{(i)}$$

We check whether $y^{(t)}\theta^{(d)} \cdot x^{(t)} > 0$ for all t to ensure there are no mistakes. Notice that the only non-zero term of the dot product occurs when $i = t$. Thus,

$$y^{(t)}\theta^{(d)} \cdot x^{(t)} = \left(y^{(t)}\right)^2 \left\|x^{(t)}\right\|^2 > 0$$

for all $t = 1, 2, \cdots, d$. Therefore, the perceptron algorithm will make exactly d updates regardless of the order and the labelings of the feature vectors.

Show answer

Submit You have used 1 of 3 attempts

Answers are displayed within the problem

Sketching convergence

0/1 point (graded)

Consider the case with $d = 3$. Also assume that all the feature vectors are positively labelled. Let P denote the plane through the three points in a 3-d space whose vector representations are given by the feature vectors $x^{(1)}, x^{(2)}, x^{(3)}$.

Let $\hat{\theta}$ denote the value of θ after perceptron algorithm converges for this example. Let v denote the vector connecting the origin and $\hat{\theta}$. Which of the following options is true regarding the vector represented by $\hat{\theta}$.

- ○ v is parallel to the plane P
- ○ v is perpendicular to the plane P and pointing away from it
- [✓ v is perpendicular to the plane P and pointing towards it]
- ○ $\hat{\theta}$ lies on the plane P

Solution:

Note that from the previous problem

$$\hat{\theta} = \sum_{i=1}^{d} y^{(i)} x^{(i)}$$

Evaluating the above expression, for the current example gives,

$$\hat{\theta} = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}$$

The equation of the plane is given by:

$$\begin{vmatrix} x & -1 & 0 \\ y-1 & -1 & -1 \\ z & 0 & -1 \end{vmatrix} = 0$$

That is,

$$x - y + z = -1$$

Or equivalently,

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \cdot \begin{pmatrix} -1 \\ 1 \\ -1 \end{pmatrix} = 1$$