16831 Homework 2 - Theory

Jennifer Isaza

TOTAL POINTS

77 / 85

QUESTION 1

1 2.1.1 Hinge loss **5 / 5**

✓ **- 0 pts** Correct

QUESTION 2

2 2.1.2 Lower bound **3 / 5**

✓ **- 2 pts** Incomplete

QUESTION 3

3 2.1.3 Upper bound **4 / 5**

✓ **- 1 pts** Incomplete explanation

How does step a imply step b

1 step a

2 step b

QUESTION 4

4 2.1.4 Chain **4 / 5**

✓ **- 1 pts** Incorrect: Proved after squaring

QUESTION 5

5 2.2.1 Multi-class hinge explanation **3 / 5**

✓ **- 1 pts** Incomplete/incorrect explanation for when learner is correct

✓ **- 1 pts** Incomplete/incorrect explanation for when learner is incorrect

Loss is not always 0 when leaner is correct (it is between 0 and 1).
Loss is >= 1 when learner is incorrect.

QUESTION 6

6 2.3 Perceptron implementation **15 / 15**

✓ **- 0 pts** Correct

QUESTION 7

7 3.1 Weight of a single point **4 / 5**

✓ **- 1 pts** Missing/incorrect/unexplained bound

QUESTION 8

8 3.2 Normalizing weights **5 / 5**

✓ **- 0 pts** Correct

QUESTION 9

9 3.3 Number of iterations **4.5 / 5**

✓ **- 0.5 pts** Error

3 Should be a lower bound

QUESTION 10

10 4.1 Hyperplanes **5 / 5**

✓ **- 0 pts** Correct

QUESTION 11

11 4.2 Subgradients **4.5 / 5**

✓ **- 0.5 pts** q4 answer does not provide correct subgradients at the origin

QUESTION 12

12 4.3 Soft SVM implementation **20 / 20**

✓ **- 0 pts** Correct

16-831 Assignment 2

jennifei

October 2018

I worked with Chris Song, Divya Kulkarni, and Andrew Wong on this assignment.

2 Perceptron

2.1 Perceptron in the Binary Non-Separable Setting

2.1.1 Hinge Loss

Given:

$$l(\mathbf{w}; (x_t, y_t)) = max\{0, 1 - y_t {x_t}^T \mathbf{w}\}$$

and

$$l(\mathbf{w}; (x_t, y_t)) \geq 1[\hat{y_t} \neq y_t] - w^T u_t$$

Plug in values:

$$max\{0, 1 - y_t {x_t}^T \mathbf{w}\} \geq 1[\hat{y_t} \neq y_t] - w^T (y_t x_t 1[\hat{y_t} \neq y_t])$$

Case 1: $\hat{y_t} \neq y_t$

$$max\{0, 1 - y_t {x_t}^T \mathbf{w}\} \geq 1 - w^T y_t x_t$$

holds true, since $1 - y_t {x_t}^T \mathbf{w} = 1 - w^T y_t x_t$, so if $0 > 1 - y_t {x_t}^T \mathbf{w}$, we also know it 0 ¿ $1 - w^T y_t x_t$
Case 2: $\hat{y_t} = y_t$

$$max\{0, 1 - y_t {x_t}^T \mathbf{w}\} \geq 0$$

holds true, since max always returns 0 if $0 > 1 - y_t {x_t}^T \mathbf{w}$

2.1.2 Lower Bound on the Potential Function

Given:

$$\phi(w_t) = {w_t}^T w^*,$$

$$L = \sum_{t=1}^{T} l(w^*; (x_t, y_t))$$

1 2.1.1 Hinge loss **5 / 5**

✓ **- 0 pts** **Correct**

16-831 Assignment 2

jennifei

October 2018

I worked with Chris Song, Divya Kulkarni, and Andrew Wong on this assignment.

2 Perceptron

2.1 Perceptron in the Binary Non-Separable Setting

2.1.1 Hinge Loss

Given:

$$l(\mathbf{w}; (x_t, y_t)) = max\{0, 1 - y_t {x_t}^T \mathbf{w}\}$$

and

$$l(\mathbf{w}; (x_t, y_t)) \geq 1[\hat{y_t} \neq y_t] - w^T u_t$$

Plug in values:

$$max\{0, 1 - y_t {x_t}^T \mathbf{w}\} \geq 1[\hat{y_t} \neq y_t] - w^T (y_t x_t 1[\hat{y_t} \neq y_t])$$

Case 1: $\hat{y_t} \neq y_t$

$$max\{0, 1 - y_t {x_t}^T \mathbf{w}\} \geq 1 - w^T y_t x_t$$

holds true, since $1 - y_t {x_t}^T \mathbf{w} = 1 - w^T y_t x_t$, so if $0 > 1 - y_t {x_t}^T \mathbf{w}$, we also know it 0 ¿ $1 - w^T y_t x_t$
Case 2: $\hat{y_t} = y_t$

$$max\{0, 1 - y_t {x_t}^T \mathbf{w}\} \geq 0$$

holds true, since max always returns 0 if $0 > 1 - y_t {x_t}^T \mathbf{w}$

2.1.2 Lower Bound on the Potential Function

Given:

$$\phi(w_t) = {w_t}^T w^*,$$

$$L = \sum_{t=1}^{T} l(w^*; (x_t, y_t))$$

$$l(\mathbf{w};(x_t,y_t)) \geq 1[\hat{y}_t \neq y_t] - w^T u_t$$

Prove: $\phi(w_{T+1}) \geq M - L$
Using ${u_t}^T w^* = \phi(w_{t+1}) - \phi(w_t)$
Rearrange and plug in:

$${u_t}^T w^* \geq -L_t + 1[\hat{y}_t \neq y_t]$$

$$\phi(w_{t+1}) = (-L_t + 1[\hat{y}_t \neq y_t]) + \phi(w_t)$$

where $M = \sum_{t=1}^{T} 1[\hat{y}_t \neq y_t]$
Induction gives:

$$\phi(w_{0+1}) = (-L_1 + M_1) + \phi(w_t)$$
$$\phi(w_{1+1}) = (-L_1 + M_1) + \phi(w_t) + M_2 - L_2$$
$$\vdots$$
$$\phi(w_{t+1}) = M - L + \phi(w_t)$$

2.1.3 Upper Bound of the Potential Function

Show: $||w_{T+1}||_2^2 \leq MR^2$
Given:

$$w_{T+1} = w_T + u_T$$

Proof:

$$||w_{T+1}||_2^2 = ||w_T + u_T||_2^2$$
$$||w_{T+1}||_2^2 = ||w_T||_2^2 + ||u_T||_2^2 + 2 < w_T, u_T >$$

where $2 < w_T, u_T >$ must be negative

$$||w_{T+1}||_2^2 = ||w_T||_2^2 + ||u_T||_2^2$$

Plug in u_T:

$$||w_{T+1}||_2^2 = ||w_T||_2^2 + ||y_T x_T 1[\hat{y}_t \neq y_t]||_2^2$$ 1
$$||w_{T+1}||_2^2 = ||w_T||_2^2 + y_T||x_T||_2^2 M$$ 2
$$||w_{T+1}||_2^2 = ||w_T||_2^2 + y_T R^2 M$$

Therefore, $||w_{T+1}||_2^2$ is linearly proportional to M and $||w_{T+1}||_2^2 \leq MR^2$

2 2.1.2 Lower bound 3 / 5

✓ - 2 pts Incomplete

$$l(\mathbf{w};(x_t,y_t)) \geq 1[\hat{y}_t \neq y_t] - w^T u_t$$

Prove: $\phi(w_{T+1}) \geq M - L$
Using ${u_t}^T w^* = \phi(w_{t+1}) - \phi(w_t)$
Rearrange and plug in:

$${u_t}^T w^* \geq -L_t + 1[\hat{y}_t \neq y_t]$$

$$\phi(w_{t+1}) = (-L_t + 1[\hat{y}_t \neq y_t]) + \phi(w_t)$$

where $M = \sum_{t=1}^{T} 1[\hat{y}_t \neq y_t]$
Induction gives:

$$\phi(w_{0+1}) = (-L_1 + M_1) + \phi(w_t)$$
$$\phi(w_{1+1}) = (-L_1 + M_1) + \phi(w_t) + M_2 - L_2$$
$$\vdots$$
$$\phi(w_{t+1}) = M - L + \phi(w_t)$$

2.1.3 Upper Bound of the Potential Function

Show: $||w_{T+1}||_2^2 \leq MR^2$
Given:

$$w_{T+1} = w_T + u_T$$

Proof:

$$||w_{T+1}||_2^2 = ||w_T + u_T||_2^2$$
$$||w_{T+1}||_2^2 = ||w_T||_2^2 + ||u_T||_2^2 + 2 < w_T, u_T >$$

where $2 < w_T, u_T >$ must be negative

$$||w_{T+1}||_2^2 = ||w_T||_2^2 + ||u_T||_2^2$$

Plug in u_T:

$$||w_{T+1}||_2^2 = ||w_T||_2^2 + ||y_T x_T 1[\hat{y}_t \neq y_t]||_2^2$$ 1
$$||w_{T+1}||_2^2 = ||w_T||_2^2 + y_T||x_T||_2^2 M$$ 2
$$||w_{T+1}||_2^2 = ||w_T||_2^2 + y_T R^2 M$$

Therefore, $||w_{T+1}||_2^2$ is linearly proportional to M and $||w_{T+1}||_2^2 \leq MR^2$

3 2.1.3 Upper bound 4 / 5

✓ **- 1 pts Incomplete explanation**

How does step a imply step b

1. step a
2. step b

2.1.4 Chain the Upper and Lower Bounds

Lower Bound: $\phi(w_{T+1}) \geq M - L$
Upper Bound: $\phi(w_{T+1}) \leq \sqrt{MR^2}$
Combining Bounds: $M - L \leq \sqrt{MR^2}$

$$M^2 - 2ML + L^2 \leq MR^2$$
$$M^2 - M(2L + R^2) + L^2 \leq 0$$
$$M \leq frac(2L + R^2) - \sqrt{4L^2 + 4LR^2 + R^4 - 4L^2}2$$
$$M \leq L - R\sqrt{L} + R\sqrt{\frac{1}{4}R^2} + \frac{1}{2}R^2$$
$$M \leq L + O(\sqrt{L}) + O(1)$$

2.2 Multi-class Perceptron

2.2.1 Understanding Multi-class Hinge Loss

Given:
$$l(W; (x_t, y_t)) = max0, 1 - (W^{y_t}x_t - maxW^j x_t)$$

If the algorithm doesn't make a mistake:
The loss function compares $W^{y_t}x_t$, which will be positive, and $maxW^j x_t$, which will be negative. This will make a max comparison of zero and a negative number, which makes the loss 0.

If this loss function upper bounds zero-one loss:
The zero-one loss can only be either zero or one and this hinge-loss can be larger than 1 when $\hat{y}_t$ is incorrect. Therefore, this hinge-loss algorithm would be an upper bound for the zero-one loss.

4 2.1.4 Chain 4 / 5

✓ **- 1 pts** **Incorrect: Proved after squaring**

2.1.4 Chain the Upper and Lower Bounds

Lower Bound: $\phi(w_{T+1}) \geq M - L$
Upper Bound: $\phi(w_{T+1}) \leq \sqrt{MR^2}$
Combining Bounds: $M - L \leq \sqrt{MR^2}$

$$M^2 - 2ML + L^2 \leq MR^2$$
$$M^2 - M(2L + R^2) + L^2 \leq 0$$
$$M \leq frac(2L + R^2) - \sqrt{4L^2 + 4LR^2 + R^4 - 4L^2}2$$
$$M \leq L - R\sqrt{L} + R\sqrt{\frac{1}{4}R^2} + \frac{1}{2}R^2$$
$$M \leq L + O(\sqrt{L}) + O(1)$$

2.2 Multi-class Perceptron

2.2.1 Understanding Multi-class Hinge Loss

Given:
$$l(W; (x_t, y_t)) = max0, 1 - (W^{y_t}x_t - maxW^j x_t)$$

If the algorithm doesn't make a mistake:
The loss function compares $W^{y_t}x_t$, which will be positive, and $maxW^j x_t$, which will be negative. This will make a max comparison of zero and a negative number, which makes the loss 0.

If this loss function upper bounds zero-one loss:
The zero-one loss can only be either zero or one and this hinge-loss can be larger than 1 when $\hat{y}_t$ is incorrect. Therefore, this hinge-loss algorithm would be an upper bound for the zero-one loss.

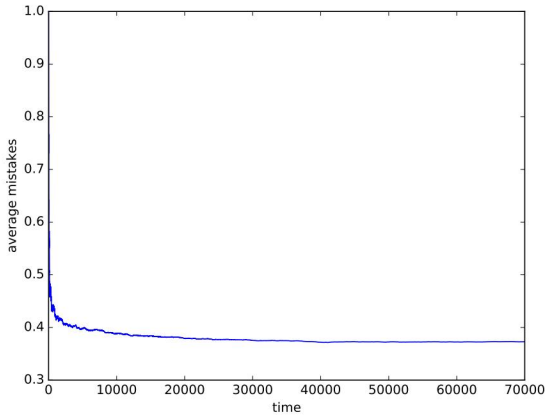
5 2.2.1 Multi-class hinge explanation 3 / 5

✓ **- 1 pts** **Incomplete/incorrect explanation for when learner is correct**

✓ **- 1 pts** **Incomplete/incorrect explanation for when learner is incorrect**

Loss is not always 0 when leaner is correct (it is between 0 and 1).
Loss is >= 1 when learner is incorrect.

2.3 Implementing Perceptron



Perceptron Loss

3 AdaBoost

3.1 Bounding the weight of a single point

Rewrite $p_{t+1}(m)$: $\frac{p_t(m)}{Z_t} e^{-y_m \sum_{t=1}^{T} \beta_t h_t(x_m)}$

$$p_1(m) = \frac{1}{M}$$

$$p_2(m) = \frac{1}{Z_1 * M} e^{-y_m \beta_1 h_1(x_m)}$$

$$p_3(m) = \frac{1}{Z_2 * Z_1 * M} e^{-y_m (\beta_1 h_1(x_m) + \beta_2 h_2(x_m))}$$

$$\vdots$$

$$p_{T+1}(m) = \frac{1}{M} \frac{e^{(-y_m \sum_{t=1}^{T} \beta_t h_t(x_m))}}{\prod_{t=1}^{T} Z_t}$$

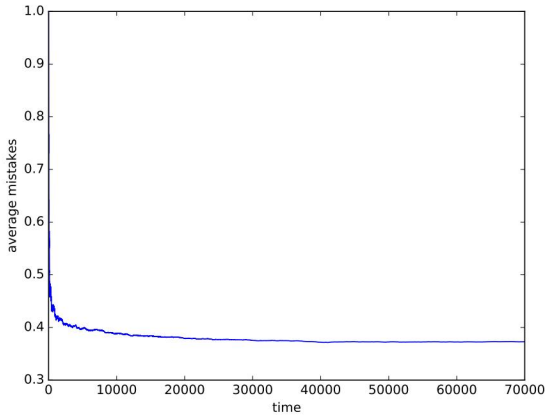
So, the lower bound of point (x_i, y_i) for which h_F get wrong is:

$$p_{T+1}(i) >= \frac{1}{M} \frac{e^{(-y_i \sum_{t=1}^{T} \beta_t h_t(x_i))}}{\prod_{t=1}^{T} Z_t}$$

6 2.3 Perceptron implementation 15 / 15

✓ - 0 pts Correct

2.3 Implementing Perceptron



Perceptron Loss

3 AdaBoost

3.1 Bounding the weight of a single point

Rewrite $p_{t+1}(m)$: $\frac{p_t(m)}{Z_t} e^{-y_m \sum_{t=1}^{T} \beta_t h_t(x_m)}$

$$p_1(m) = \frac{1}{M}$$

$$p_2(m) = \frac{1}{Z_1 * M} e^{-y_m \beta_1 h_1(x_m)}$$

$$p_3(m) = \frac{1}{Z_2 * Z_1 * M} e^{-y_m (\beta_1 h_1(x_m) + \beta_2 h_2(x_m))}$$

$$\vdots$$

$$p_{T+1}(m) = \frac{1}{M} \frac{e^{(-y_m \sum_{t=1}^{T} \beta_t h_t(x_m))}}{\prod_{t=1}^{T} Z_t}$$

So, the lower bound of point (x_i, y_i) for which h_F get wrong is:

$$p_{T+1}(i) >= \frac{1}{M} \frac{e^{(-y_i \sum_{t=1}^{T} \beta_t h_t(x_i))}}{\prod_{t=1}^{T} Z_t}$$

7 3.1 Weight of a single point **4 / 5**

✓ **- 1 pts** **Missing/incorrect/unexplained bound**

3.2 Bounding error using normalizing weights

$$\epsilon = \frac{1}{M}\sum_{m=1}^{M} \mathbb{1}(y_m \neq h_F(x_m))$$

When $y_m = h_F(x_m)$:

$y_m h_F(x_m) > 0$, so $\mathbb{1}[y_m \neq h_F(x_m)]$ can be rewritten as $\mathbb{1}(y_m h_F(x_m) <= 0)$

So, we can rewrite ϵ as:

$$\epsilon = \frac{1}{M}\sum_{m=1}^{M} \mathbb{1}[y_m h_F(x_m) <= 0]$$

Using the relationship $\mathbb{1}[\alpha \leq 0] \leq e^{-\alpha}$:

$$\epsilon = \frac{1}{M}\sum_{m=1}^{M} \mathbb{1}[y_m h_F(x_m) <= 0] \leq \frac{1}{M}\sum_{m=1}^{M} e^{y_m h_F(x_m)}$$

Notice that we can get the right side equation into the form of $p_{T+1}(m)$ if we plug in $h_F(x) = sign(\sum_{m=1}^{M} \beta_t h_t(x_m))$ and divide both sides by $\prod_{t=1}^{T} Z_t$:

$$\frac{1}{M}\frac{\sum_{m=1}^{M} \mathbb{1}[y_m h_F(x_m) <= 0]}{\prod_{t=1}^{T} Z_t} \leq \frac{1}{M}\frac{e^{-y_m \sum_{m=1}^{M} \beta_t h_t(x_m)}}{\prod_{t=1}^{T} Z_t} = p_{T+1}(m)$$

Using the property: $\sum_m p(m) = 1$:

$$\frac{1}{M}\frac{\sum_{m=1}^{M} \mathbb{1}[y_m h_F(x_m) <= 0]}{\prod_{t=1}^{T} Z_t} \leq 1$$

$$\epsilon = \frac{1}{M}\sum_{m=1}^{M} \mathbb{1}[y_m h_F(x_m) <= 0] \leq \prod_{t=1}^{T} Z_t$$

$$\epsilon \leq \prod_{t=1}^{T} Z_t$$

3.3 How many iterations should you run AdaBoost?

$$\epsilon \leq 2^T \prod_{t=1}^{T} \sqrt{\epsilon_t(1-\epsilon_t)} < \epsilon_0$$

$$2^T \prod_{t=1}^{T} \sqrt{(\frac{1}{2}-\gamma)(1-(\frac{1}{2}-\gamma))} < \epsilon_0$$

$$2^T \prod_{t=1}^{T} \sqrt{\frac{1}{4}-\gamma^2} < \epsilon_0$$

8 3.2 Normalizing weights 5 / 5

✓ - 0 pts Correct

3.2 Bounding error using normalizing weights

$$\epsilon = \frac{1}{M}\sum_{m=1}^{M} \mathbb{1}(y_m \neq h_F(x_m))$$

When $y_m = h_F(x_m)$:

$y_m h_F(x_m) > 0$, so $\mathbb{1}[y_m \neq h_F(x_m)]$ can be rewritten as $\mathbb{1}(y_m h_F(x_m) <= 0)$

So, we can rewrite ϵ as:

$$\epsilon = \frac{1}{M}\sum_{m=1}^{M} \mathbb{1}[y_m h_F(x_m) <= 0]$$

Using the relationship $\mathbb{1}[\alpha \leq 0] \leq e^{-\alpha}$:

$$\epsilon = \frac{1}{M}\sum_{m=1}^{M} \mathbb{1}[y_m h_F(x_m) <= 0] \leq \frac{1}{M}\sum_{m=1}^{M} e^{y_m h_F(x_m)}$$

Notice that we can get the right side equation into the form of $p_{T+1}(m)$ if we plug in $h_F(x) = sign(\sum_{m=1}^{M} \beta_t h_t(x_m))$ and divide both sides by $\prod_{t=1}^{T} Z_t$:

$$\frac{1}{M}\frac{\sum_{m=1}^{M} \mathbb{1}[y_m h_F(x_m) <= 0]}{\prod_{t=1}^{T} Z_t} \leq \frac{1}{M}\frac{e^{-y_m \sum_{m=1}^{M} \beta_t h_t(x_m)}}{\prod_{t=1}^{T} Z_t} = p_{T+1}(m)$$

Using the property: $\sum_m p(m) = 1$:

$$\frac{1}{M}\frac{\sum_{m=1}^{M} \mathbb{1}[y_m h_F(x_m) <= 0]}{\prod_{t=1}^{T} Z_t} \leq 1$$

$$\epsilon = \frac{1}{M}\sum_{m=1}^{M} \mathbb{1}[y_m h_F(x_m) <= 0] \leq \prod_{t=1}^{T} Z_t$$

$$\epsilon \leq \prod_{t=1}^{T} Z_t$$

3.3 How many iterations should you run AdaBoost?

$$\epsilon \leq 2^T \prod_{t=1}^{T} \sqrt{\epsilon_t(1-\epsilon_t)} < \epsilon_0$$

$$2^T \prod_{t=1}^{T} \sqrt{(\frac{1}{2}-\gamma)(1-(\frac{1}{2}-\gamma))} < \epsilon_0$$

$$2^T \prod_{t=1}^{T} \sqrt{\frac{1}{4}-\gamma^2} < \epsilon_0$$

Can take $\sqrt{(\frac{1}{2}-\gamma)}$ out of product, since not dependent on time:

$$2^T\sqrt{\frac{1}{4}-\gamma^2}\prod_{t=1}^{T}1<\epsilon_0$$

$$log_2(2^T\sqrt{\frac{1}{4}-\gamma^2})<log_2(\epsilon_0)$$

$$T+log_2(\sqrt{\frac{1}{4}-\gamma^2})<log_2(\epsilon_0)$$

$$T<log_2\left(\frac{\epsilon_0}{\sqrt{\frac{1}{4}-\gamma^2}}\right)$$ 3

4 Soft SVM

4.1 Understanding Hyperplanes

1.) A hyperplane in 2D is a line of the form wx + b = 0, where w is a weight in the 2D real space. A hyperplane in 3D is a plane of the form wx + b = 0, where w is a vector in the 3D real space.
2.) $w=(2,1,1), x=(x_1,x_2,1)$:

9 3.3 Number of iterations **4.5 / 5**

✓ **- 0.5 pts** **Error**

3 Should be a lower bound

Can take $\sqrt{(\frac{1}{2}-\gamma)}$ out of product, since not dependent on time:

$$2^T\sqrt{\frac{1}{4}-\gamma^2}\prod_{t=1}^{T}1<\epsilon_0$$

$$log_2(2^T\sqrt{\frac{1}{4}-\gamma^2})<log_2(\epsilon_0)$$

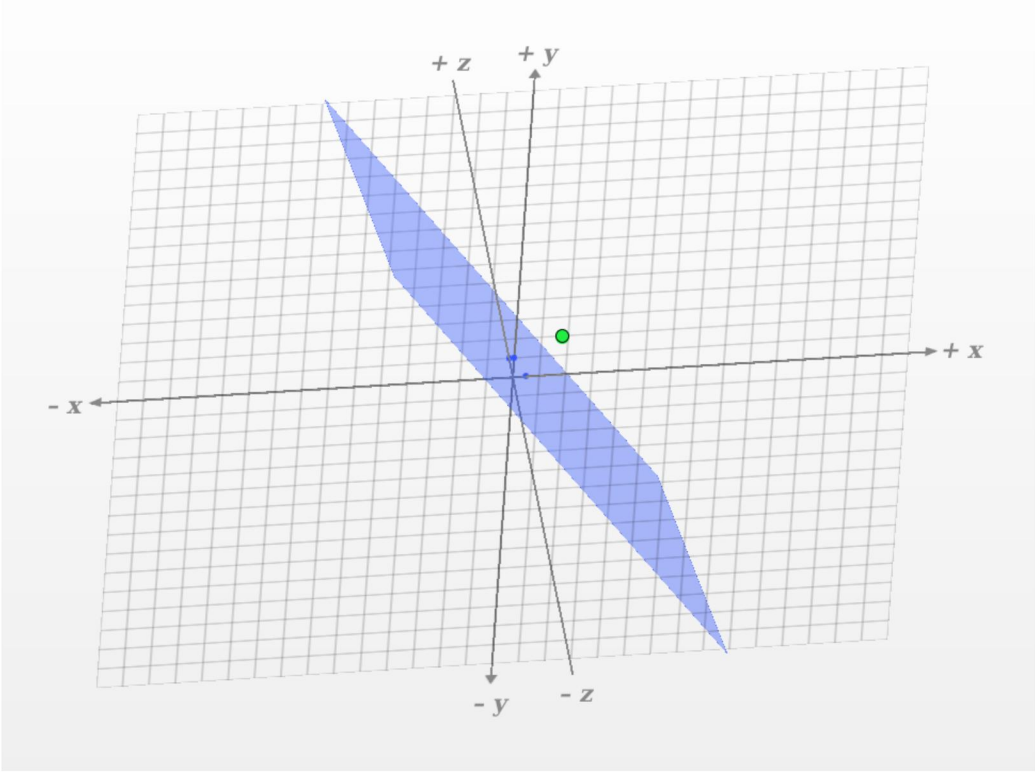
$$T+log_2(\sqrt{\frac{1}{4}-\gamma^2})<log_2(\epsilon_0)$$

$$T<log_2\left(\frac{\epsilon_0}{\sqrt{\frac{1}{4}-\gamma^2}}\right)$$ 3

4 Soft SVM

4.1 Understanding Hyperplanes

1.) A hyperplane in 2D is a line of the form wx + b = 0, where w is a weight in the 2D real space. A hyperplane in 3D is a plane of the form wx + b = 0, where w is a vector in the 3D real space.
2.) $w=(2,1,1), x=(x_1,x_2,1)$:



Hyperplane Normal to (2,1,1)

3.) On the plane: $x = \begin{bmatrix} 5 \\ 4 \\ 1 \end{bmatrix}$ Other sides of plane: $x = \begin{bmatrix} 5 \\ 4 \\ 0 \end{bmatrix}$ and $x = \begin{bmatrix} 5 \\ 4 \\ 2 \end{bmatrix}$

4.) The normal vector of theis hyperplane is w. The distance between the normal vector and the origin is $\frac{b}{||w||}$

5.) We end up with a parallel but different hyperplane if a normalization factor of $\frac{1}{20}$ is used on $w = \begin{bmatrix} 40 \\ 20 \\ 1 \end{bmatrix}$ because b would also be normalized and would no longer be equal to 1, like it was for $w = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix}$

4.2 Subgradients

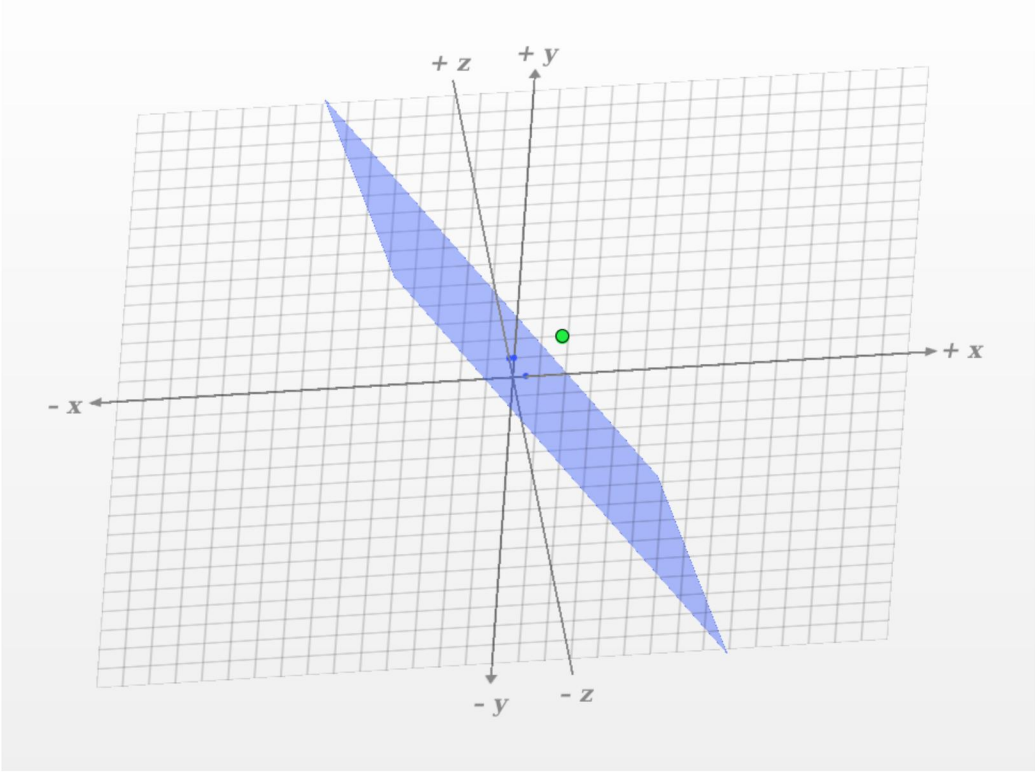
1.) The hinge loss is positive for when y_i and $w_i x$ have the same sign and negative for when y_i and $w_i x$ have different signs. It penalizes input that is further from the hyperplane.

2.) Yes, hinge loss is convex with respect to the w vector of weights.

3.) No, the hinge loss is not differentiable.

10 4.1 Hyperplanes **5 / 5**

✓ **- 0 pts** **Correct**



Hyperplane Normal to (2,1,1)

3.) On the plane: $x = \begin{bmatrix} 5 \\ 4 \\ 1 \end{bmatrix}$ Other sides of plane: $x = \begin{bmatrix} 5 \\ 4 \\ 0 \end{bmatrix}$ and $x = \begin{bmatrix} 5 \\ 4 \\ 2 \end{bmatrix}$

4.) The normal vector of theis hyperplane is w. The distance between the normal vector and the origin is $\frac{b}{||w||}$

5.) We end up with a parallel but different hyperplane if a normalization factor of $\frac{1}{20}$ is used on $w = \begin{bmatrix} 40 \\ 20 \\ 1 \end{bmatrix}$ because b would also be normalized and would no longer be equal to 1, like it was for $w = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix}$

4.2 Subgradients

1.) The hinge loss is positive for when y_i and $w_i x$ have the same sign and negative for when y_i and $w_i x$ have different signs. It penalizes input that is further from the hyperplane.

2.) Yes, hinge loss is convex with respect to the w vector of weights.

3.) No, the hinge loss is not differentiable.

4.) The two subgradients for the hinge loss are:

0 if $y_m w^T x_m \geq$ to 1

$-y_m x_m$ otherwise

4.3 Implementing the Soft SVM

1.) Algorithm's accuracy for each class:
Class 1004: 840/1000 correct
Class 1100: 972/1000 correct
Class 1103: 683/1000 correct
Class 1200: 938/1000 correct
Class 1400: 854/1000 correct
Multiclass: 806/1000 correct

2.) I chose λ to be 1, so that theta would update by smaller and smaller amounts as time increased. This is based off an assumption that the updates will push theta in the generally correct direction at the start of training. I did not change my training or testing size, (kept them at 1000), since I realized that my results were not significantly better after this increased time.

3.) The algorithm takes classes*iterations*2+iterations to train and predict.

11 4.2 Subgradients 4.5 / 5

✓ **- 0.5 pts** q4 answer does not provide correct subgradients at the origin

4.) The two subgradients for the hinge loss are:

0 if $y_m w^T x_m \geq$ to 1

$-y_m x_m$ otherwise

4.3 Implementing the Soft SVM

1.) Algorithm's accuracy for each class:
Class 1004: 840/1000 correct
Class 1100: 972/1000 correct
Class 1103: 683/1000 correct
Class 1200: 938/1000 correct
Class 1400: 854/1000 correct
Multiclass: 806/1000 correct

2.) I chose λ to be 1, so that theta would update by smaller and smaller amounts as time increased. This is based off an assumption that the updates will push theta in the generally correct direction at the start of training. I did not change my training or testing size, (kept them at 1000), since I realized that my results were not significantly better after this increased time.

3.) The algorithm takes classes*iterations*2+iterations to train and predict.

12 4.3 Soft SVM implementation 20 / 20

✓ - 0 pts Correct