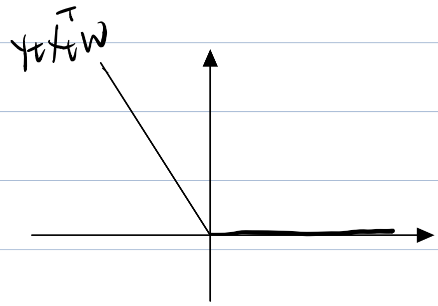
Robo Stats hw2

Zheyao Zhu

October 2020

2.1.1 Hinge loss.



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

$$u_t = y_t x_t \mathbb{1}[\hat{y}_t \neq y_t]$$

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

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

when $\hat{y}_t \neq y_t$

$x_t^T w < 0$

$\Downarrow$

left side : $1 - y_t x_t^T w$ $\Rightarrow$ left side = right side.

right side : $1 - y_t x_t^T w$

when $\hat{y}_t = y_t$

$x_t^T w > 0$

$\Downarrow$

if $0 < y_t x_t^T w < 1$:

left side $= 1 - y_t x_t^T w > 0$ $\Rightarrow$ left = right

right side : $(1 - y_t x_t^T w) > 0$.

if $y_t x_t^T w > 1$

left side $= 0$ $\Rightarrow$ left $>$ right.

right side. < 0

if $y_t x_t^T w = 1$

left $= 0$ $\Rightarrow$ left $=$ right

right $= 0$

Thus proved.

2.1.2 Lower Bound on the Potential Function

Need to prove that $\Phi(\omega_{T+1}) \geq M - L$

$=> \omega_{T+1}^T \omega^* \geq \sum_{t+1}^T 1[\hat{y_t} \neq y_t] - \sum_{t+1}^T l(\omega^*, (x_t, y_t))$

At T=1
Left side: $\omega_2^T \omega^* = \omega_1^T \omega^* + \omega^{*T} u_1 = 0 + \omega^{*T} u_1$

Right side: $1[\hat{y_1} \neq y_1] - l(\omega^*, (x_1, y_1))$

Since we have prove in the previous question that:
$l(\omega^*, (x_t, y_t)) \geq 1[\hat{y_t} \neq y_t] - \omega^{*T} u_t$

We know that:
$l(\omega^*, (x_1, y_1)) \geq 1[\hat{y_1} \neq y_1] - \omega^{*T} u_1$

$=> \omega^{*T} u_1 \geq 1[\hat{y_1} \neq y_1] - l(\omega^*, (x_1, y_1))$

=> Left side $\geq$ Right side

Thus proved

Assume it is true at **T= n**

$=> \omega_{n+1}^T \omega^* \geq \sum_{n+1}^T 1[\hat{y_n} \neq y_n] - \sum_{n+1}^T l(\omega^*, (x_n, y_n))$

$=> \omega_n^T \omega^* + \omega^{*T} u_n \geq \sum_{n+1}^T 1[\hat{y_n} \neq y_n] - \sum_{n+1}^T l(\omega^*, (x_n, y_n))$

At **T = n+1**
Left side = $\omega_n^T \omega^* + \omega^{*T} u_n + \omega^{*T} u_{n+1}$

Right side = $\sum_{n+1}^T 1[\hat{y_n} \neq y_n] - \sum_{n+1}^T l(\omega^*, (x_n, y_n)) + 1[\hat{y_{n+1}} \neq y_{n+1}] - l(\omega^*, (x_{n+1}, y_{n+1}))$

Subtracting both sides with expression from T = n

=> Left side = $\omega^{*T} u_{n+1}$

Right side = $1[\hat{y_{n+1}} \neq y_{n+1}] - l(\omega^*, (x_{n+1}, y_{n+1}))$

Since we have prove in the previous question that:
$l(\omega^*, (x_t, y_t)) \geq 1[\hat{y_t} \neq y_t] - \omega^{*T} u_t$

We can prove that
$\omega^{*T} u_t \geq 1[\hat{y_t} \neq y_t] - l(\omega^*, (x_t, y_t))$

Thus we have proven that $\Phi(\omega_{T+1}) \geq M - L$ by induction

2.1.3 Upper Bound on the Potential Function

Need to prove that $||W_{T+1}||_2^2 \leq MR^2$

When T=1
Left side: $||W_2||_2^2 = ||W_1 + y_1 x_1 1[\hat{y_t} \neq y_t]||_2^2 = ||y_1 x_1 1[\hat{y_t} \neq y_t]||_2^2$

if mistake: left = = $||x_1||_2^2$ right = R^2

=> Left < right by definition

If no mistake: left = right = 0

Assume it is true when T= n
=> $||W_{+1}||_2^2 \leq MR^2$

=> $||W_n + y_n x_n 1[\hat{y_n} \neq y_n]||_2^2 \leq \sum_{t=1}^{t=n} 1[\hat{y_n} \neq y_n] R^2$

When T = n+1

Left: $||W_n + y_n x_n 1[\hat{y_n} \neq y_n] + y_{n+1} x_{n+1} 1[\hat{y_{n+1}} \neq y_{n+1}]||_2^2$

Right: $(\sum_{t=1}^{t=n} 1[\hat{y_n} \neq y_t] + 1[\hat{y_{n+1}} \neq y_{n+1}]) R^2$

if $\hat{y_{n+1}} = y_{n+1}$

Left $= ||W_n + y_n x_n 1[\hat{y_n} \neq y_n]||_2^2$

Right $= \sum_1^n 1[\hat{y_n} \neq y_n] R^2$

Left $\leq$ Right from T=n

Subtract the expression from both sides with expression of both sides when T=n results in

Left $= ||y_{n+1} x_{n+1} 1[\hat{y_{n+1}} \neq y_{n+1}]||_2^2$

Right $= 1[\hat{y_n} \neq y_n] R^2$
if $\hat{y_{n+1}} \neq y_{n+1}$
we then need to prove $||X_{n+1}|| \leq R^2$. It is proved by definition

Thus we have proved that $||W_{T+1}||_2^2 \leq MR^2$
Z

2.1.4 Chain the upper and lower bound

From previous question we know that

$\Phi(W_{t+1}) \leq \sqrt{M}RD$ and $M - L \leq \Phi(W_{t+1})$

$=> M - L \leq \Phi(W_{t+1}) \leq \sqrt{M}RD$

$=> M^2 - 2ML + L^2 \leq MR^2D^2$

$=> M^2 - M(2L + R^2D^2) + L^2 \leq 0$

Using quadratic equation and eliminating the negative root we get:

$M - \frac{2L+R^2D^2+\sqrt{(2L+R^2D^2)^2-4L^2}}{2} \leq 0$

$M \leq \frac{2L+R^2D^2+\sqrt{(2L+R^2D^2)^2-4L^2}}{2}$

$M \leq \frac{2L+R^2D^2+\sqrt{4LR^2D^2+(R^2D^2)^2}}{2}$

$$M \leq L + \frac{\sqrt{4LR^2D^2 + (R^2D^2)^2}}{2} + \frac{R^2D^2}{2}$$

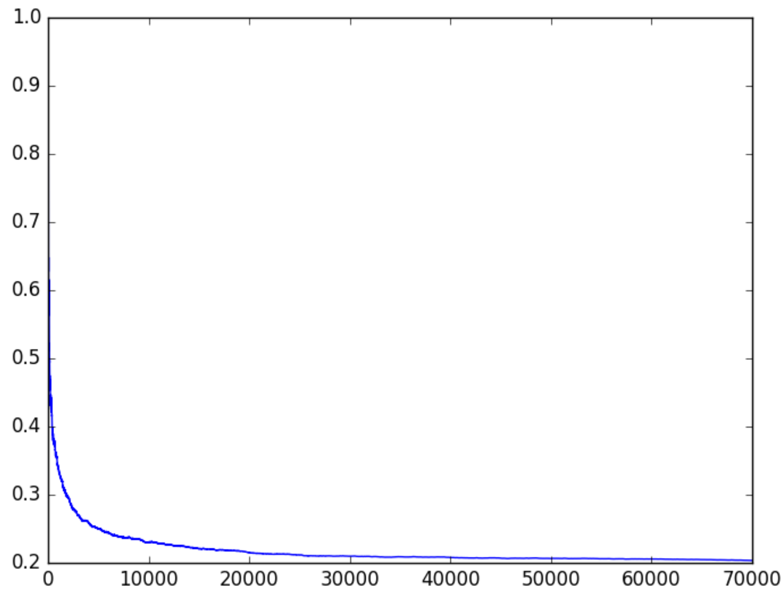
$$M \leq L + 0(\sqrt{L}) + 0(1)$$

2.2 Multi-class Perceptron

2.2.1 Understanding Multi Class Hinge Loss

It makes sense because $W^{y_t}x_t$- max$W^j x_t$ represents the loss of the classifier that the predictor is based on since the predictions are made by $\hat{y}_t = argmax_{j\in\{1,2,3,..k\}} W^j x_t$

2.3 Implementing Perceptron



3 Adaboost

3.1 Bounding the weight of a single point

The reweighting step could be written as:

$P_{t+1}(m) = \frac{P_t(m)}{Z_t} exp(-\beta_t y_m h_t(x_m))$

Since $y_m h_t(x_m) = -1$ if $y_m \neq h_t(x_m)$ and $y_m h_t(x_m) = 1$ if $y_m = h_t(x_m)$

When T =1

$P_2(m) = \frac{1}{m} \cdot \frac{1}{z} exp(-\beta_1 h_1(x+m) y_m)$

When T= 2

$P_3(m) = \frac{1}{m} \frac{1}{z_1} exp(-\beta_1 h_!(x_m) y_m) \cdot (\frac{1}{z_2} exp(-\beta_2 h_2(x_m) y_m))$

$= \frac{1}{m} \frac{1}{z_1 \cdot z_3} exp(-\beta_1 h_1(x_m) + \beta_2 h_2(x_m) y_m)$

$=> p_{(t+1)m} = \frac{1}{m} \frac{1}{\prod_1^T z_t} exp(\sum_1^T \beta_t h_t(x_m) y_m)$

$=> p_{(t+1)m} = \frac{1}{m} \frac{1}{\prod_1^T z_t} exp(-y_m f(x_m))$

Find a lower bound for $P_{T+1}(i) of point (x_i, y_i) which h_F gets wrong$

if h_F gets it wrong then $-y_m f(x_m) > 0$ which means $exp(y_m f(x_m)) \geq 1$

3.2 Bounding error using normalizing weights

$\epsilon(t) = \sum_1^M p_t(m) \cdot 1(y_m \neq h_t(x_m))$

$\leq \sum_1^M p_t(m) exp(-y_i f(x_i))$

$= exp(-y_i \sum_1^T \beta_t h_t(x))$

$= \prod_{t=1}^T exp(-y_i \beta_t h_t(x))$

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

3.3 Choosing β_t to minimize error bound

In order to minimize error bound we need to choose *beta* such that

$\beta_t = \frac{1}{n} \ln(\frac{1-\epsilon_t}{\epsilon_t})$

$Z_t = \sum_{i:y_i \neq h_t(x_i)} D_t(i) e^{\alpha_t} + \sum_{i:y_i = h_t(x_i)} D_t(i) e^{-\alpha_t}$

$$= \epsilon_t e^{\alpha_t} + (1 + \epsilon_t) e^{-\alpha_t}$$

$$= 2\sqrt{\epsilon_t (1 - \epsilon_t)} = \sqrt{1 - (1 - 2\epsilon_t)^2}$$

3.4. How many iterations to run Adaboost.

$$e^{2\beta_t} = \frac{1-\epsilon_t}{\epsilon_t}$$

$\epsilon_t = \frac{1}{2} - \gamma_t$ such that $\gamma_t \geq \gamma$

$$\epsilon = \frac{1}{M}\sum_{m=1}^{M} \mathbb{1}(y_m \neq h_f(x_m)) \leq 2^T \prod_{t=1}^{T} \sqrt{\epsilon_t(1-\epsilon_t)}$$

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

use inequality $1-x \leq \exp(-x)$

$$\leq \prod_t^T \sqrt{\exp(2\epsilon_t-1)^2}$$

$$\leq \exp \sum_1^T \frac{1}{2}(2\epsilon_t-1)^2$$

$$\leq \exp \sum_1^T \frac{1}{2}\cdot 4(\epsilon_t - \frac{1}{2})^2$$

$$\leq \exp 2\sum_1^T (\epsilon_t - \frac{1}{2})^2$$

Assume $1/2 - \gamma_t = 1/2 - \gamma$

$$\Rightarrow \varepsilon \leq \exp(-2\sum_1^T \gamma_t^2) = \exp(-2T\gamma^2)$$

$$\Rightarrow T \leq \frac{1}{2\gamma^2}\ln\frac{1}{\varepsilon}$$

4 Soft SVM

4.1.1 Existence of Subgridents

It is false. For instance, consider the function $y = -x^2$
if you take a derivative of the function at any point, the derivative would upperbound instead of lowerbound the function. Thus wouldn't be considered as subgradient.

4.1.2 Finding subgradients

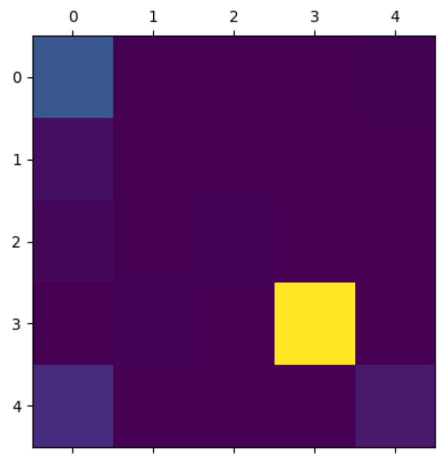
1.
if $f_1(x) > f_2(x)$, then $\frac{df}{dx} = \frac{df_1(x)}{dx}$
if $f_2(x) > f_1(x)$, then $\frac{df}{dx} = \frac{df_2(x)}{dx}$

2.
$f(x) = ||x||_2 = \sqrt(\sum_i x_i^2) = (\sum_i x_i^2)^{\frac{1}{2}}$

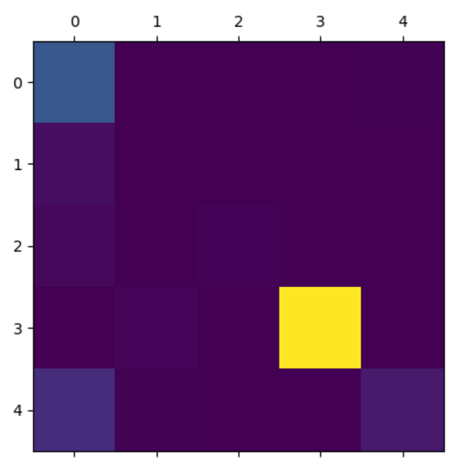
$\frac{df(x)}{dx} = \frac{1}{2}(\sum_i x_i^2)^{-\frac{1}{2}} \cdot \sum_i 2x_i$

4.2 Implementing the soft SVM

1. Confusion Matrix
Confusion Matrix with training data



Confusion Matrix with testing data

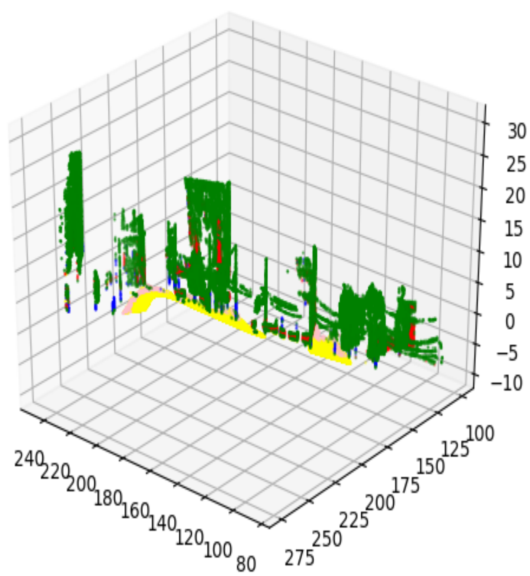


2. Model Performance
The trained model was getting 86.14% accuracy on training data and 85.78% accuracy on testing data

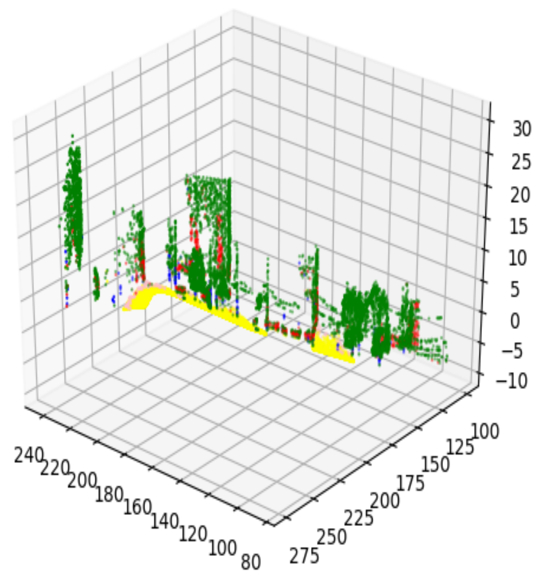
The model performed especially well on the ground and veg while it performed poorly on wire and poles. The reason I think is that the wire are really close to the trees so it's hard to linearly separate wire and trees. In addition, wires are connected to the poles so they can't be linearly separated either. As a result, the poles and wires are all classified as vegetation.

3.Point Cloud
Color:
Veg: Green
Wire: Pink
Pole: Blue
Ground: Yellow
Facade: Red

Classified pointcloud of training data



Classified pointcloud of testing data



4.Choice of hyperparameter
The only hyperparameter I had was the learning rate. I experimented with a couple learning rate and found that learning rate of one works just fine.

5.training
I was able to get decent result by training the model with only one epoch which is one path through all the data in the training data. I found that increasing training epoch didn't really increase the performance.