CS246 Homework 4 Answers

Charlie Zhang

Mar 2013

1 Question 1 –Support Vector Machine

1.1 (a)

The example:

(0, 0): -1

(0, 1): -1

(1, 0): 1

(1, 1): 1

(2, 0): -1

This is unfeasible under hard constraints SVM but feasible under soft margin SVM.

Such as: $w = (2, 0), b = -1, \xi_1, .., \xi_5 = (0, 0, 0, 0, 4)$

1.2 (b)

Lets set:

$w_j = 0, \forall j = 1, ..., d,$

$b = 0,$

$\xi_i = 1, \forall i = 1, ..., n$

Then $y_i(w \cdot x + b) \geq 1 - \xi_i$ holds true for all i.

1.3 (c)

Let T be the training set and E be the set of points where linear classification mis-classified.

Then $y_i(x_i \cdot w + b) < 0, \forall i \in E$.

Also, $(w, \xi_i, ..., \xi_n)$ is a feasible point, so $y_i(x_i \cdot w + b) \geq 1 - \xi_i, \forall i \in T$ Now we have $1 - \xi_i < 0$, i.e., $\xi_i > 1$, $\forall i \in E$

So $\sum\limits_{i \in T} \xi_i >= \sum\limits_{i \in E} \xi_i > |E|$

1.4 (d)

$\nabla_b f(w, b) = \frac{\delta f(w,b)}{\delta b} = C \sum\limits_{i=1}^{n} \frac{\delta L(x_i, y_i)}{\delta b}$

Where $\frac{\delta L(x_i,y_i)}{\delta b} = \begin{cases} 0 & \text{if } y_i(w \cdot x_i + b) \geq 1 \\ -y_i & \text{otherwise} \end{cases}$

1.5 (e)

Red: BatchGradientDescend

Green: StochasticGradientDescend

Blue: MiniBatchGradientDescend

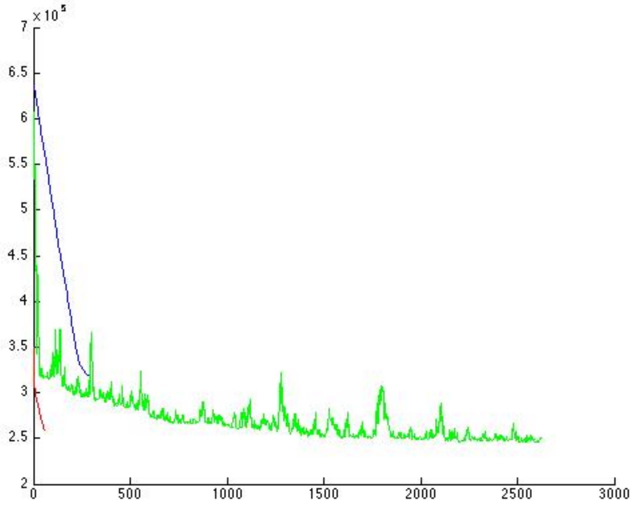


Figure 1: Cost as of iterations.

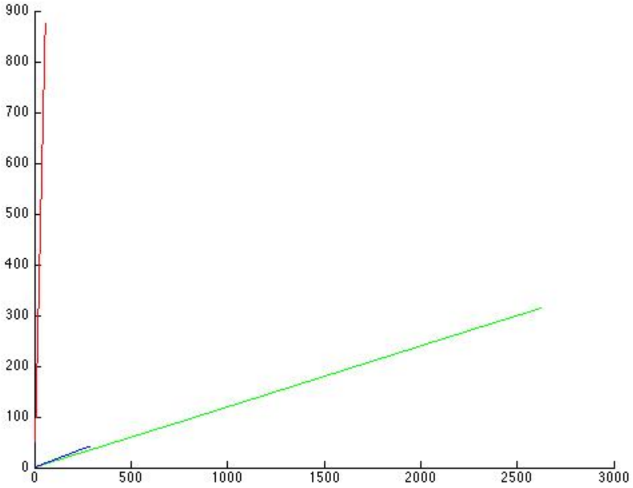


Figure 2: Time elapsed as of iterations.

What to infer:

Stochastic Gradient Descend makes many small but quick steps, some of the steps are wrong, but overall, it converges faster than Batch Gradient Descend in terms of computation time. But given sufficient computation resource, Batch Gradient Descend could achieve better result than Stochastic Gradient Descend. Mini Batch Gradient Descend is a mixure of the two methods above, it makes more steps and runs faster than Batch Gradient Descend, the cost is also larger. It runs slower than Stochastic Gradient Descend, but it makes less error steps.

1.6 (f)

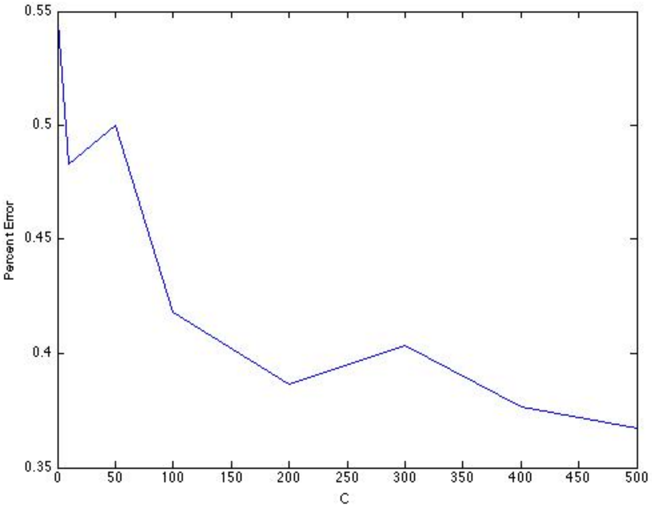


Figure 3: $\boldsymbol{C}$ Versus Percent Error.

Conclusion:

Percent error gets smaller as $\boldsymbol{C}$ gets larger.

2 Question 2 –Decision Tree Learning

2.1 (a)

$G = max[I(D) - (I(D_L) + I(D_R))]$

There we only consider one attribute, so:

$G = I(D) - (I(D_L) + I(D_R)) =$

$|D| \times (1 - \sum_i p_i^2) - |D_L| \times (1 - \sum_i p_{L(i)}^2) - |D_R| \times (1 - \sum_i p_{R(i)}^2) =$

$|x+y+u+v| * ((1 - \frac{(x+u)^2}{(x+y+u+v)^2}) + (1 - \frac{(y+v)^2}{(x+y+u+v)^2})) - |x+y| * ((1 - \frac{x^2}{x+y)^2}) +$

$(1 - \frac{y^2}{x+y)^2})) - |u+v| * ((1 - \frac{u^2}{u+v)^2}) + (1 - \frac{v^2}{u+v)^2})) =$

$\frac{x^2+y^2}{x+y} + \frac{u^2+v^2}{u+v} - \frac{(x+u)^2+(y+v)^2}{x+y+u+v} > 0$

Solve the inequality and we get $(xv - yu)^2 > 0$.

So $\frac{x}{y} \neq \frac{u}{v}$.

2.2 (b)

based on the equation in (a), we have:

$G_{wine} = \frac{30^2+20^2}{50} + \frac{30^2+20^2}{50} - \frac{60^2+40^2}{100} = 0$

$G_{running} = \frac{20^2+10^2}{30} + \frac{40^2+30^2}{70} - \frac{60^2+40^2}{100} = 0.381$

$G_{pizza} = \frac{50^2+30^2}{80} + \frac{10^2+10^2}{20} - \frac{60^2+40^2}{100} = 0.500$

We should use the attribute 'likes pizza'

2.3 (c)

The decision tree is a complete binary tree with a depth of 100.

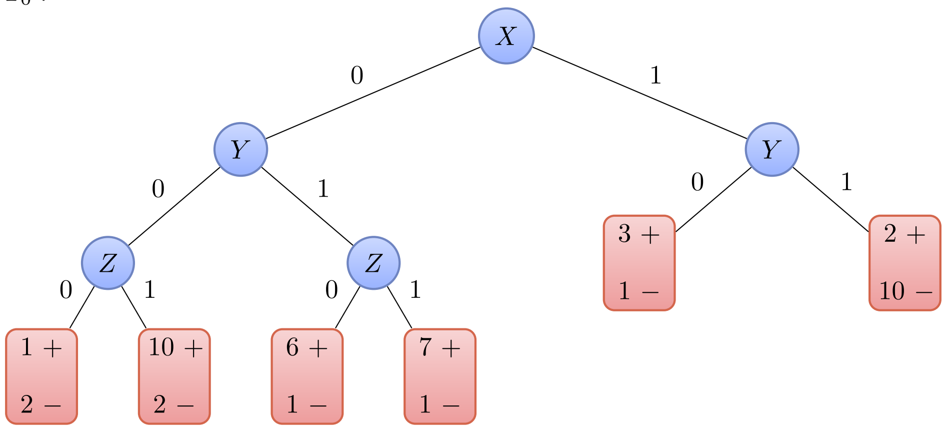
There are 2^{100} leaf nodes, which are the decision nodes, corresponding to the $\{0,1\}^{100}$ feature space.

The root is a_1, the two branches are $a_1 = 0$ and $a_1 = 1$ respectively, the rest of the tree (2nd layer to leaf) are nodes for $a_2 ... a_{100}$, the arrangement of $a_2 ... a_{100}$ in these nodes are arbitrary, depending on the features of those elements whose target value not equal to a_1.

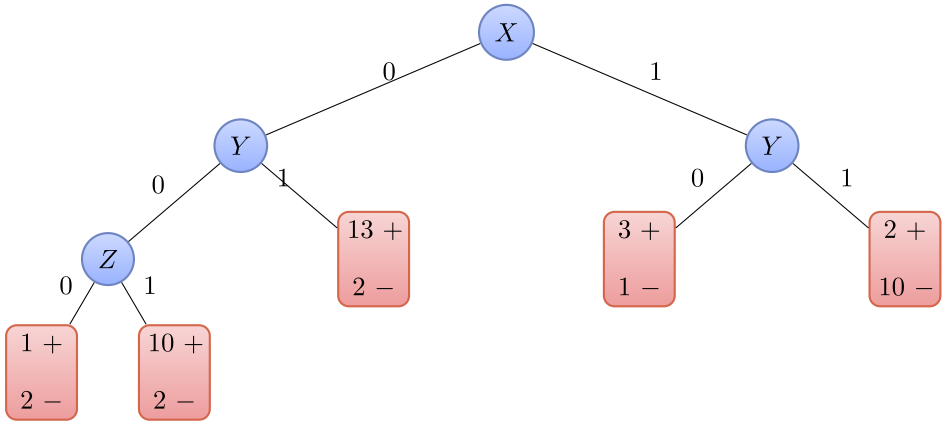
A desired decision tree should have a depth of 1, and 2 leaf nodes.
The decision criterion are $a_1 = 0$ and $a_1 = 1$.
This tree will not over fit to those 1% training data whose target value not equals to a_1.

2.4 (d)

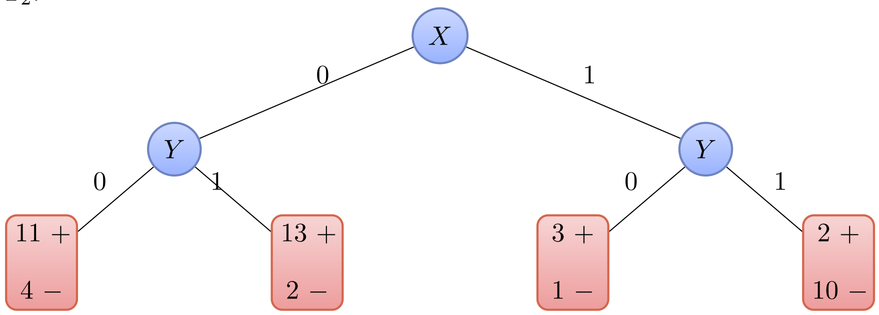
T_0 :



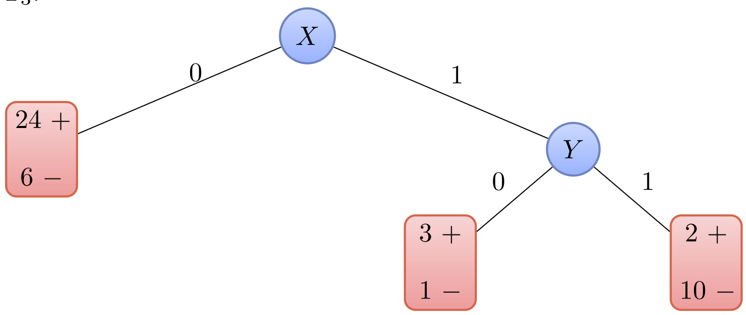
T_1:



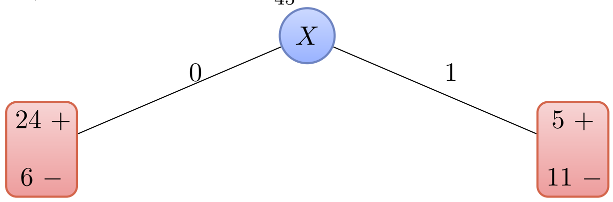
T_2:



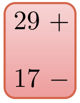
T_3:



T_4, obtained with $\alpha = \frac{2}{45}$:



T_5:



2.5 (e)

Generalization errors for $\{T_0 \ldots T_5\}$ are:

$\{\frac{2}{4}, \frac{2}{4}, \frac{1}{4}, \frac{1}{4}, 0, \frac{2}{4}\}$.

T_4 is the one with best generalization error.

3 Question 3 –Clustering Data Streams

3.1 (a)

$2a^2 + 2b^2 - (a+b)^2 =$

$2a^2 + 2b^2 - (a^2 + 2ab + b^2) =$

$a^2 - 2ab + b^2 = (a-b)^2 \geq 0.$

So $(a+b)^2 \leq 2a^2 + 2b^2$.

3.2 (b)

Based on conclusion in (a), we have $2d(x,y)^2 + 2d(y,T)^2 \geq (d(x,y) + d(y,T))^2 \geq d(x,T)^2, \forall x, y \in S, \forall T$.

$2 \cdot cost_w(\hat{S}, T) + 2\sum_{i=1}^{l} cost(S_i, T_i) =$

$2\sum_{i=1}^{l}\sum_{j=1}^{k} |S_{ij}| d(t_{ij}, T)^2 + 2\sum_{i=1}^{l}\sum_{j=1}^{k}\sum_{x \in S_{ij}} d(x, t_{ij})^2 =$

$\sum_{i=1}^{l}\sum_{j=1}^{k}\sum_{x \in S_{ij}} (2d(x, t_{ij})^2 + 2d(t_{ij}, T)^2) \geq$

$\sum_{i=1}^{l}\sum_{j=1}^{k}\sum_{x \in S_{ij}} d(x, T)^2 =$

$cost(S, T)$

3.3 (c)

$\sum_{i=1}^{l} cost(S_i, T_i) \leq \alpha \sum_{i=1}^{l} cost(S_i, T_i^*)$, where T_i^* is the optimal solution for S_i

Since T_i^* is the optimal solution for S_i, $cost(S_i, T_i^*) \leq cost(S_i, T^*)$.

$\sum_{i=1}^{l} cost(S_i, T_i^*) \leq \sum_{i=1}^{l} cost(S_i, T^*) = cost(S, T^*)$

So $\sum_{i=1}^{l} cost(S_i, T_i) \leq \alpha \cdot cost(S, T^*)$.

3.4 (d)

According to the definition of ALG, $cost_w(S,T) \leq \alpha \min_{|T'|=k} \{cost_w(S,T')\}$. T is obtained by running ALG on $\hat{S}$, by definition, we have:

$cost_w(\hat{S}, T) \leq \alpha cost_w(\hat{S}, T^*)$

3.5 (e)

$2\sum_{i=1}^{l} cost(S_i, T_i) + 2cost(S, T^*) = 2\sum_{i=1}^{l}\sum_{j=1}^{k}\sum_{x \in S_{ij}} d(x, t_{ij})^2 + 2\sum_{x \in S} d(x, T^*)^2 =$

$\sum_{i=1}^{l}\sum_{j=1}^{k}\sum_{x \in S_{ij}} 2(d(x, t_{ij})^2 + d(x, T^*)^2) \geq$

$\sum_{i=1}^{l}\sum_{j=1}^{k}\sum_{x \in S_{ij}} (d(x, t_{ij}) + d(x, T^*))^2 \geq$

$\sum_{i=1}^{l}\sum_{j=1}^{k} |S_{ij}| d(t_{ij}, T^*)^2) =$

$cost_w(\hat{S}, T^*)$

So, $cost_w(\hat{S}, T^*) \leq 2\sum_{i=1}^{l} cost(S_i, T_i) + 2cost(S, T^*)$

3.6 (f)

Based on conclusions in (b), (c), (d), (e), we have:

$cost(S,T) \leq 2 \cdot cost_w(\hat{S}, T) + 2\sum_{i=1}^{l} cost(S_i, T_i) =$

$2\alpha \cdot cost_w(\hat{S}, T^*) + 2\alpha \cdot cost(S, T^*) \leq$

$2\alpha \cdot (2\sum_{i=1}^{l} cost(S_i, T_i) + 2cost(S, T^*)) + 2\alpha \cdot cost(S, T^*) \leq$

$2\alpha \cdot (2\alpha \cdot cost(S, T^*) + 2cost(S, T^*)) + 2\alpha \cdot cost(S, T^*) =$

$(4\alpha^2 + 6\alpha) \cdot cost(S, T^*)$

3.7 (g)

Assumes that we know the value of n in the first place.

Set number of partition $l = \sqrt{\frac{n}{k}}$, then, $|S_i| = \frac{|S|}{l} = \sqrt{nk}, \forall i = 1, ..., l.$

Maximum space requirement =

Space for storing S_i and run ALG on S_i plus space for storing T, and space for running ALG on T =

max((space of ALG for S_i) $+kl$, space of ALG for $\hat{S}$)

= max($\frac{n}{l}$ + kl, kl)

= $\sqrt{nk}$

With choices of $l = \sqrt{\frac{n}{k}}$, assuming that we know n ahead,

ALGSTR works with O($\sqrt{nk}$) space.

4 Question 4 – Data Streams

4.1 (a)

Every occurrence of i, we increment the count $c_{j,h_j(i)}$, $\forall j \in [[1; [log\frac{1}{\delta}]]]$.

Suppose there are no other elements than i in the stream, then $c_{j,h_j(i)} = F[i]$, $\forall j \in [[1; [log\frac{1}{\delta}]]]$

In this case $min_j\{c_{j,h_j(i)}\} = F[i]$

Since there could be other elements in the stream that collides with i in hashed values, so overall, $\tilde{F}[i] = min_j\{c_{j,h_j(i)}\} \geq F[i]$

4.2 (b)

Since h_j maps {1,2,...n} to $\{1, 2, ...[\frac{e}{\epsilon}]\}$ uniformly, the expected number of other elements falls in $c_{j,h_j(i)}$ for hash function j is:
$\frac{(t-F[i])}{[\frac{e}{\epsilon}]}$.
So, for any $1 \leq i \leq n$ and $1 \leq j \leq [log\frac{1}{\delta}]$, $E[c_{j,h_j}] = F[i] + \frac{(t-F[i])}{[\frac{e}{\epsilon}]} \leq F[i] + \frac{\epsilon}{e}(t - F[i])$.

4.3 (c)

$Pr[\tilde{F}[i] \leq F[i] + \epsilon t] = 1 - (Pr[c_{j,h_j(i)} > F[i] + \epsilon t])^{[log\frac{1}{\delta}]}$

Based on Markov's Inequality,

$Pr[c_{j,h_j(i)} > F[i] + \epsilon t] =$

$Pr[c_{j,h_j(i)} - F[i] > \epsilon t] \leq$

$\frac{E[c_{j,h_j(i)}-F[i]]}{\epsilon t} \leq$

$\frac{F[i]+\frac{\epsilon}{e}(t-F[i])-F[i]}{\epsilon t} \leq \frac{\frac{\epsilon}{e}t}{\epsilon t} = \frac{1}{e}$ So $Pr[\tilde{F}[i] \leq F[i] + \epsilon t] =$
$1 - (Pr[c_{j,h_j(i)} > F[i] + \epsilon t])^{[log\frac{1}{\delta}]} \geq 1 - \frac{1}{e}^{log\frac{1}{\delta}} = 1 - \delta$

4.4 (d)

In HW3-Q4, we need $O(n)$ extra memory to keep track of the degree of each node, where n is the number of nodes.
With the approach provided above, we can bring down the extra memory requirement from $O(n)$ to $O(\frac{1}{\epsilon}log[\frac{1}{\delta}])$.