Problem Set 4 for Machine Learning 15 Fall

Jingyuan Liu
AndrewId: jingyual
jingyual@andrew.cmu.edu

1 VC dimension

1.1 Show the VC dimension of linear classifier

To prove that the linear classifier h with x in R^n has the VC dimension of n + 1, we need to prove that $VCdim(h_n) >= n+1$, and then prove that $VCdim(h_n) <= n+1$.

(a). Prove $VCdim(h_n) >= n+1$

First of all, we could use the Mathematical Induction to prove that $VCdim(H) >= n+1$:

For n = 1, it is easy to get $VCdim(h_1) >= 2$

For n = 2, we could also get $VCdim(h_2) >= 3$, which could be proved using following figures:

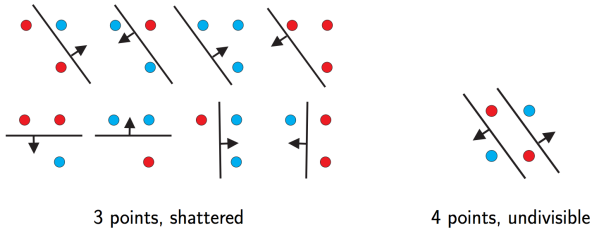

Figure 1: The representation of $VCdim(h_2)$

For n = i, we assume that the $VCdim(h_i) >= i+1$. We could form a matrix of (i+1)*(i+1), with the rank of i+1, since the VCdim is i+1.

For n = i+1, we could form a matrix of (i+2)*(i+2), since the variable x have one more independent dimension. Therefore, $VCdim(h_{i+1}) >= VCdim(h_i) + 1 >= i+2$.

Therefore, we have $VCdim(h_n) >= n+1$. We could use a more straight forward detailed example to show that. Say we have n dimensions of the data, then we could construct a dataset:

$1th$: (1,0,0,...,0)

$2th$: (0,1,0,...,0)

ith: (0,0,...,i..,0)

nth: (0,0,0,...,n)

$n+1th$: (0,0,....,0)

For these dataset, we could see that for the ith data point, the ith dimension is 1, otherwise 0. And for the n+1th data, all dimension are 0. In this case, we could see that the linear classifier could shatter this dataset.

(b). Prove $VCdim(h_n) <= n + 1$

Then we should prove that $VCdim(h_n) <= n + 1$:

Suppose we could shatter n+2 points, then using the convex combinations of points in C, we could seperate points into S1 and S2, that:

$$conv(s1) \cap conv(s2) \neq \emptyset \tag{1}$$

However, if $VCdim(h_n) = n + 2$, then we could spilt points to half space contains s1 and the complement of the half space contains s2. This implies that the both half space contains the convex hull of s1 and the complement of the half space the contains the convex hull of s2. Thus, we get:

$$conv(s1) \cap conv(s2) = \emptyset \tag{2}$$

These two observations are contradictory to each other. So we know that for the n+2 points here, we could not use the linear classifier to shatter all of them.

In conclusion, we prove that $VCdim(h_n) >= n + 1$ and $VCdim(h_n) <= n + 1$ for the linear classifier, then we have $VCdim(h_n) = n + 1$

1.2 Show the VC dimension of axis-aligned boxes

Similarily, for this case, to prove that the axis-aligned boxes classifier h with x in R^n has the VC dimension of 2n, we need to prove that $VCdim(h_n) >= 2n$, and then prove that $VCdim(h_n) <= 2n$.

(a). Prove $VCdim(h_n) >= 2n$

First of all, we need to prove that for any case, $VCdim(h_n) >= 2n$, which means if the x has n dimensions, we can use the axis-aligned boxes classifiers to shatter 2n points.

Suppose we have n dimensions, then we can map all the x to a dimension i. In the dimension i, we can have x^i_{max} and x^i_{min}. Therefore, in this dimension i, we could always shatter at least 2 points via a split between the x^i_{max} and x^i_{min}.

Considering we have n dimensions, and the mapping of each dimension of the data is independent to the mapping of other dimension of the data. Therefore, we could at least shatter 2*n points via the axias-alighed boxes. Therefore, we have $VCdim(h_n) >= 2n$.

(b). Prove $VCdim(h_n) <= 2n$

Then we need to prove that $VCdim(h_n) <= 2n$. Suppose we have 2n + 1 points, we could always find the 2n "boundries", which is the min value and max value for each dimension, and form an "area". Then the 2n+1th points will be guaranteed to appear within the selected "area".

For any 2n+1 points, we could always transfer to the above mentioned scenario. In this scenario, we could not classify the 2n+1th point, because it is contained within the "area", and no rules are guaranteed to rightly classify it.

Therefore, by mapping the data to each dimension and form an "area", we could prove that $VCdim(h_n) <= 2n$.

In conclusion, we prove that $VCdim(h_n) >= 2n$ and $VCdim(h_n) <= 2n$ for the axis-alighed boxes, then we have $VCdim(h_n) = 2n$

2 AdaBoost

2.1 Justify the Update Rule

Based on the defination, if we change the distribution, the previous learned hypothesis h_t would be a "random guess" to the new distribution of data, therefore:

$$\epsilon_t = err_{D_t}(h_t) = Pr_{x\sim D_t}(y \neq h_t(x)) = \frac{1}{2} \tag{3}$$

The error rate is $\frac{1}{2}$ for a random guess with margin γ_t = 0. Then we would have, for all i:

$$\alpha_t = \frac{1}{2} log(\frac{1-\epsilon_t}{\epsilon_t}) = 0 \tag{4}$$

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} e^{-\alpha_t y_i h_t(x_i)} \tag{5}$$

$$= \frac{D_t(i)}{Z_t} \tag{6}$$

Therefore, we could see that the $D_{t+1} = D_t$, so:

$$err_{D_{t+1}}(h_t) = Pr_{x\sim D_t}(y \neq h_t(x)) \tag{7}$$

$$= \sum_{y_i \neq h_t(x_i)} D_{t+1}(i) \tag{8}$$

$$= \sum_{y_i \neq h_t(x_i)} D_t(i) \tag{9}$$

$$= err_{D_t} = \frac{1}{2} \tag{10}$$

2.2 Show the $D_{t+1}(i)$

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} e^{-\alpha_t y_i h_t(x_i)} \tag{11}$$

$$= \frac{e^{-y_i}}{Z_t} e^{\alpha_t h_t(x_i)} D_t(i) \tag{12}$$

$$= \frac{e^{-y_i}}{Z_t} e^{\alpha_t h_t(x_i)} \cdot \frac{e^{-y_i}}{Z_{t-1}} e^{\alpha_{t-1} h_{t-1}(x_i)} D_{t-2}(i) \tag{13}$$

$$= \frac{e^{-y_i}}{Z_t} e^{\alpha_t h_t(x_i)} \cdots \frac{1}{m} e^{\alpha_1 h_1(x_i)} \tag{14}$$

$$= (m \prod_t Z_t)^{-1} e^{-y_i \sum_t \alpha_t h_t(x)} \tag{15}$$

2.3 Show the error upperbound

$$err_S(H) = Pr(y \neq H(x)) = \frac{1}{m} \sum_{H(i) \neq y_i} D_S(i) \tag{16}$$

When $H(i) \neq y_i$, we have $y_i f(x_i) < 0$, and $exp(-y_i f(x_i)) > 0$, therefore:

$$err_S(H) < \frac{1}{m} \sum_i exp(-y_i f(x_i)) \tag{17}$$

For the right term, according to the defination of Z and W, we have:

$$w_{t,i} exp(-\alpha_t y_i h_t(x_i)) = Z_m w_{t+1,i} \tag{18}$$

$$\frac{1}{m} \sum_i exp(-y_i f(x_i)) = \frac{1}{m} \sum_i exp(-\sum_t \alpha_t y_i h_t(x_i)) \tag{19}$$

$$= \sum_{1,i} w_{1,i} \prod_t exp(-\alpha_t y_i h_t(x_i)) \tag{20}$$

$$= Z_1 \sum_{2,i} w_{2,i} \prod_{t=2}^{T} exp(-\alpha_t y_i h_t(x_i)) \tag{21}$$

$$= Z_1 Z_2 \sum_{3,i} w_{3,i} \prod_{t=3}^{T} exp(-\alpha_t y_i h_t(x_i)) \tag{22}$$

$$= Z_1 Z_2 \cdots \sum_i w_{T,i} exp(\alpha_T y_i H(x_i)) \tag{23}$$

$$= \prod_t Z_t \tag{24}$$

Therefore, we have:

$$err_S(H) < \frac{1}{m} \sum_i exp(-y_i f(x_i)) = \prod_t Z_t \tag{25}$$

2.4 Show the $\prod_t Z_t$ upperbound

$$Z_t = \sum_i exp(-\alpha_t y_i h_t(x_i)) \tag{26}$$

$$= \sum_{h_t(x_i)=y_i} w_i e^{-\alpha_t} + \sum_{h_t(x_i) \neq y_i} w_i e^{\alpha_t} \tag{27}$$

$$= (1 - \epsilon_t) e^{-\alpha_t} + \epsilon e^{\alpha_t} \tag{28}$$

$$= 2\sqrt{\epsilon_t (1 - \epsilon_t)} = \sqrt{1 - 4\gamma_t^2} \tag{29}$$

Therefore, we could use Taylor expansion series to get $\sqrt{1 - 4\gamma_t^2} <= exp(-2\gamma_t^2)$, so we have:

$$\prod_t Z_t <= e^{-2 \sum_t \gamma_t^2} \tag{30}$$

2.5 Express big-O notation of T

From above questions, we could get:

$$\epsilon <= e^{-2\sum_t \gamma_t^2} \tag{31}$$

With the smallest margin of γ, we could transfer to

$$\epsilon <= e^{-2T\gamma^2} \tag{32}$$

$$T = O(\frac{-log(\epsilon)}{\gamma^2}) \tag{33}$$

2.6 implementation

For each h, we choose the classifier with smallest error in each iteration. The result is:

t	ϵ_t	α_t	$D_t(1)$	$D_t(2)$	$D_t(3)$	$D_t(4)$	$D_t(5)$	$D_t(6)$	$D_t(7)$	$D_t(8)$	$D_t(9)$	$err_S(H)$
1	0.222	0.626	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.11	0.222
2	0.143	0.896	0.071	0.071	0.071	0.071	0.071	0.071	0.071	0.25	0.25	0.222
3	0.125	0.973	0.042	0.042	0.042	0.042	0.25	0.25	0.042	0.146	0.146	0

Table 1: AdaBoost results

The classfication rule: first, if $x > 2.5$, pos; second, if $x > 3.5$, pos; third, if $x < 4.5$, pos.

3 Gaussian Mixture Model

3.1 Show the expectation

$$E(x) = \int p(x)dx = \sum_k p(x_k)x_k \tag{34}$$

$$= \sum_k \pi_k E_k(x) \tag{35}$$

We know that for each k, the implict distribution of x is a Gaussian distribution, and $E_k(x) = \mu_k$, μ_k is the mean of the kth guassian distribution, so we have:

$$E(x) = \sum_k \pi_k \mu_k \tag{36}$$

3.2 Show the covariance

$$cov(x) = E(xx^T) - E(x)E(x)^T \tag{37}$$

$$= \sum_k \pi_k E_k(xx^T) - E(x)E(x)^T \tag{38}$$

$$= \sum_k \pi_k(\Sigma_k + \mu_k\mu_k^T) - E(X)E(x)^T \tag{39}$$

4 K-Means

4.1 Theory

4.1.1 Prove the lemma

$$\sum_x \|x-s\|^2 - \sum_x \|x-\bar{x}\|^2 = \sum_x (x^2 - 2xs + s^2) - \sum_x (x^2 - 2x\bar{x} + \bar{x}^2) \tag{40}$$

$$= \sum_x s^2 + 2x\bar{x} - \bar{x}^2 - 2\bar{x}s \tag{41}$$

Considering that $\bar{x}$ is the center of x points, we have $\sum_x x\bar{x} = |X|\,\bar{x}^2$. Therefore:

$$\sum_x s^2 + 2x\bar{x} - \bar{x}^2 - 2\bar{x}s = \sum_x s^2 + \bar{x}^2 - 2\bar{x}s = |\mathcal{X}|\,\|\bar{x}-s\|^2 \tag{42}$$

4.1.2 Prove the objective

We could first transfer $w(\mu_k, f; X)$ use n_k to represent all nodes in kth cluster:

$$w(\mu_k, f; X) = \sum_k \sum_i^n 1(f(x_i) = k)\,\|x_i - \mu_k\|^2 \tag{43}$$

$$= \sum_k \sum_i^{n_k} \|x_{ki} - \mu_k\|^2 \tag{44}$$

$$= \sum_k \sum_i^{n_k} \frac{1}{n_k} n_k\,\|x_{ki} - \mu_k\|^2 \tag{45}$$

Then we consider the form from ϕ, for the kth cluster and set the i, we have:

$$\sum_j^{n_k} \|x_{ki} - x_{kj}\|^2 = \sum_j^{n_k} x_{ki}^2 - 2x_{ki}x_{kj} + x_{ki}^2 \tag{46}$$

Using lemma1 and similar tricks in proving lemma1, then we could have:

$$n_k\,\|x_{ki} - \mu_k\|^2 = \sum_j \|x_{ki} - x_{kj}\|^2 \tag{47}$$

Seperately integrating into the ϕ and w, we have:

$$w(\mu_k, f; X) = \sum_k \sum_i^n 1(f(x_i) = k)\,\|x_i - \mu_k\|^2 \tag{48}$$

$$= \sum_k \sum_i^{n_k} \sum_j^{n_k} \frac{1}{n_k} \|x_{ki} - x_{kj}\|^2 = \phi \tag{49}$$

4.1.3 Prove the decrease of objective

Proving the decrease of objective during each iteration is quite intuive:

For step 1, suppose we have two cluster μ_1 and μ_2. Suppose μ_1 is closer. Then when we assign the point:

$$\|x - \mu_1\|^2 < \|x - \mu_2\|^2 \tag{50}$$

Therefore, to choose the minimize the total objective w, we should choose μ_1, which means that step 1 will decrease the objective.

For step 2, we have lemma1:

$$\sum_x \|x - s\|^2 - \sum_x \|x - \bar{x}\|^2 = |X| \, \|\bar{x} - s\|^2 > 0 \tag{51}$$

So we know that, chooseing any other point rather than the "average" as center would cause bigger error, which means that using the "average" will decrease the objective.

4.1.4 Prove the convergence with K

Proving the convergence of $\Omega(K)$ is quite intuitive. When we have the minimual objective, then in step 1, all the x_i would not change its assignment of cluster centain. Say we have the μ and any other μ', we would have since that Ω is the minimual objective.

$$\|x - \mu\|^2 < \|x - \mu'\|^2 \tag{52}$$

For step 2, since all the x_i would not change its assignment, then the $\mu_k = \frac{1}{n_k} \sum_i x_{ki}$ would not change. So the center would remain the same.

Therefore, in step 1 and step 2, there would be no change in the assignment and center, the Ω would not change with the increase of K

4.1.5 Prove the convergence is finite

The K-means would converge in finite numbers. We know that the center is the average of all x in a cluster. When we set the K, then the combination of K is a finite number. In the step 1, the assignment step, the choice of K is finite. Then in step 2, the change of center is set.

As mentioned from Wikipedia, the time is $O(n^{dk+1} logn)$, with k as the cluster, d as the dimension of data x and n as the data points number.

4.2 Implementation

4.2.1 and 4.2.2 Implement the K-means algorithms

The result is here:

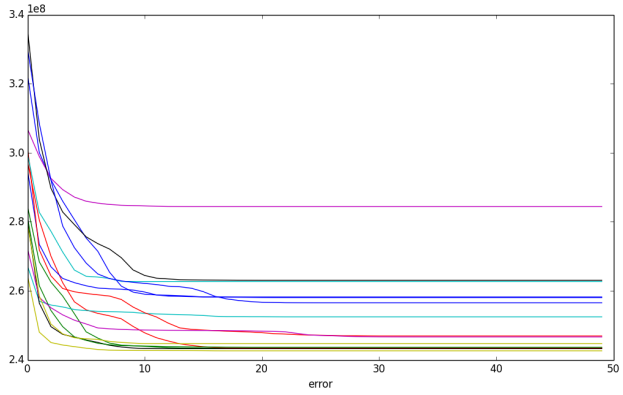

Figure 2: The error over iteration of K-means

As we could see from the figure, all the 15 iterations converge within 50 iteration. For most instances, the alogrithm would converge within $15 \sim 20$ iterations.

5 Collaboration

I did not collaborate with classmates in this assignment. However, I refered to some code online to finish my K-means algorithms. The link is: https://github.com/stuntgoat/kmeans.git

Basically, I used the general code structure, the update and assign function. I implemented my own data read and error plot function. Besides, I change the interface of the general kmeans function for stop conditions. I also implement the error function to get the error in each iteration for each cluster.