MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Department of Electrical Engineering and Computer Science
6.036—Introduction to Machine Learning
Spring Semester 2015

Assignment 5, Issued: Friday, April. 24

Not For Submission

VC Dimension

1. Prove that the VC dimension of $(d-1)$-dimensional hyperplanes through the origin (i.e. the set $\mathcal{H}$ of linear classifiers in $\mathbb{R}^d$ with no offset) is d.

Recall that for $h_\theta \in \mathcal{H}$, $h_\theta(x) = sign(x \cdot \theta)$, where x and θ are d-dimensional vectors.

Part 1) Proof that the VC dimension of $\mathcal{H}$ is at least d:

We need to show that $\exists$ a set of data points $x^{(1)}, ..., x^{(d)}$ such that $\forall$ labelings $y^{(1)}, ..., y^{(d)}$, we could find the parameters θ of a linear classifier in $\mathcal{H}$ to correctly classify these examples (i.e. $x^{(1)}, ..., x^{(d)}$ can be shattered by $\mathcal{H}$). In other words, given our selection of data points, an adversary would not be able to find a labeling that we could not to classify correctly with some $h_\theta \in \mathcal{H}$.

Let x_i denote the i-th canonical vector in $\mathbb{R}^d$ (i.e. a vector with a one in the i-th coordinate and zeros elsewhere). Claim: $x^{(1)}, ..., x^{(d)}$ can be shattered by $\mathcal{H}$. Given any $y^{(1)}, ..., y^{(d)}$, if we set $\theta_i = y^{(i)}$, then for all $x^{(i)}$, it follows that $(x^{(i)} \cdot \theta)y^{(i)} = (x_i^{(i)}\theta_i)y^{(i)} = (x_i^{(i)}y^{(i)})y^{(i)} = (y^{(i)})^2 > 0$. Thus, each $x^{(i)}$ will be correctly classified by h_θ (with our choice of θ) for any label $y^{(i)}$.

Part 2) Proof that the VC dimension of $\mathcal{H}$ is at most d:

We need to show that $\forall$ set of data points $x^{(1)}, ..., x^{(d+1)}$, $\exists$ some labeling $y^{(1)}, ..., y^{(d+1)}$ that we could not correctly classify (for any choice of θ). To show this, we need to prove that the label for $x^{(d+1)}$ is a function of the labels of $x^{(1)}, ..., x^{(d)}$ and thus can not be arbitrarily chosen. Thus, if we know $y^{(1)}, ..., y^{(d)}$, then $y^{(d+1)}$ can only take on one label (value), and thus the opposite label would be misclassified.

A collection of $d+1$ points in $\mathbb{R}^d$ must be linearly dependent. Suppose, w.l.o.g. that $x^{(d+1)}$ can be expressed as a linear combination of $x^{(1)}, ..., x^{(d)}$, then:

$$x^{(d+1)} \cdot \theta = \left(\sum_{j=1}^{d} \alpha_j x^{(j)} \right) \cdot \theta = \sum_{j=1}^{d} \alpha_j (x^{(j)} \cdot \theta)$$

whose sign is determined by the values of the other $x^{(j)}$, so we can find a labeling of $x^{(1)}, ..., x^{(d)}$ such that we can not correctly label $x^{(d+1)}$.
Consider a labeling given by $y^{(j)} = sign(\alpha_j), j = 1, \ldots, d$ and $y^{(d+1)} = -1$. Let's show that under this labeling, if we correctly classify the first d points, then we would misclassify $x^{(d+1)}$. To see this, note that:

$$x^{(d+1)} \cdot \theta = \sum_{j=1}^{d} \alpha_j (x^{(j)} \cdot \theta) \geq 0$$

which holds because each of the elements of the sum is positive: $x^{(j)} \cdot \theta$ has the same sign as $y^{(j)}$ in order for $x^{(j)}$ to be correctly classified, and by the given labeling, it has to equal the sign of α_j. Therefore, $x^{(d+1)}$ will be misclassified. Thus we can not shatter these $d+1$ points. The same argument works for any set of $d+1$ points.

By parts (1) and (2), the VC dimension of $\mathcal{H}$ is exactly d.

BIC

2. You are consulting for a trading company, BIC Inc. They are currently modeling stock values with model M_1. They are considering switching to another, more complex, model M_2, which has 10 times as many parameters as their current model. This model seems to better fit the data they have available. Given that you have taken 6.036, BIC Inc. has hired you to decide whether they should switch to model 2.

   Assume the data consists of n points, and M_1 has p parameters. How much of a better fit of the data should M_2 provide in order for you to recommend a switch? Please provide an expression for the minimal difference in data likelihood ($l_2 - l_1$) that would justify this change.

   From the BIC criterion, we have that

$$\begin{aligned}\text{BIC}(M_1) &= l_1 - \frac{1}{2}p\log n\\ \text{BIC}(M_2) &= l_2 - \frac{1}{2}p_2\log n = l_2 - \frac{1}{2}10p\log n\end{aligned}$$

   We would switch to M_2 if and only if $\text{BIC}(M_2) - \text{BIC}(M_1) > 0$, which implies

$$l_2 - l_1 > \frac{9p}{2}\log n$$

K-means and K-medoids

3. Euclidean distance is not the only way to measure the distance between two d-dimensional vectors, there is a related family of distance measures, known as l_p norms, parameterized by $p \geq 1$.

   The l_p norm of a vector is defined as: $\|x\|_p = \left(\sum_j |x_j|^p\right)^{1/p}$.

   The standard Euclidean distance is the l_2 norm of vector difference between the two points,
   $\|x - y\|_2 = \left(\sum_j |x_j - y_j|^2\right)^{1/2}$

   The Manhattan distance is the l_1 norm defined as
   $\|x - y\|_1 = \sum_j |x_j - y_j|$

   The "maximum distance" is the l_∞norm defined as
   $\|x - y\|_\infty = \max_j |x_j - y_j|$

   Assume we have a two dimensional dataset consisting of (-5,2),(4,4),(0,-6), (0,0). Assume that we set $k = 3$, and we initalize the cluster centers with $(-5, 2), (4, 4), (0, -6)$.

   At convergence, what will the cluster centers be and where will the points be assigned after we run the:

   (a) K-means algorithm with L_2 norm

   First we will assign (-5,2) and (0,0) to cluster 1, (4,4) to cluster 2, (0,-6) to 3. Then, we update the cluster means to be (-2.5,1), (4,4) (0,-6). At this point, we will have converged.

   (b) K-medoids algorithm with L_1 norm

   First we will assign (-5,2) to cluster 1, (4,4) to cluster 2, (0,-6), (0,0) to 3. Then, we update the cluster means to be (-5,2), (4,4), and (0,0). At this point, we will have converged.

   (c) K-medoids algorithm with L_2 norm

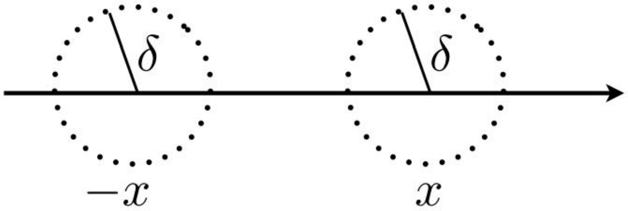
   First we will assign (-5,2) and (0,0) to cluster 1, (4,4) to cluster 2, (0,-6) to 3. Then, we update the cluster means to be (-5,2) , (4,4), and (0,-6). At this point, we will have converged.

   (d) K-medoids algorithm with L_∞ norm

   First we will assign (-5,2) to cluster 1, (4,4) and (0,0) to cluster 2, (0,-6) to 3. Then, we update the cluster means to be (-5,2) , (4,4), and (0,-6). At this point, we will have converged.

K-means Part 2

4. Consider the simple K-means algorithm for clustering. Each iteration of the algorithm consists of two steps, assigning points to the centroids, and updating the centroids based on the points assigned to them. We will assume that $k = 2$. Please give detailed explanations on your conclusion. Assume that the two clusters are well-seperated. (see the plot below, well separated means $\delta << x$)



(a) If we initialize the centroids by drawing a random point from each of the two well separated clusters, how many iterations does it take for the k-means to converge?
It will take only one step. After that, the cluster centers will be the circles of the clusters.

(b) Are there initializations such that at convergence, the K-means algorithm will not seperate the two clusters. If so, what are they?
Consider the initialization $(0, 2\delta)$, $(0, -2\delta)$. This will assign the top halves of both circles to one cluster, and the bottom halves to another.

EM algorithm

5. Consider the following mixture of two Gaussians: $P(x;\theta) = p_1 N(x;\mu_1,\sigma_1^2) + p_2 N(x;\mu_2,\sigma_2^2)$ which has adjustable parameters $\theta = \{p_1, p_2, \mu_1, \mu_2, \sigma_1^2, \sigma_2^2\}$ (the means, variances, and weights of each Gaussian). We initialize the mixture weights θ^0 using $p_1 = p_2 = 0.5, \mu_1 = 6, \mu_2 = 7, \sigma_1 = 1, \sigma_2 = 2$.

   We have the following samples of x: $x^{(0)} = 0, x^{(1)} = 1, x^{(2)} = 5, x^{(3)} = 6, x^{(4)} = 7$.

   (a) What is the loglikelihood, $l(x;\theta)$, that we are maximizing? Why do we use the EM algorithm? Does it always give the optimal solution?

   We are finding θ that maximizes following term:

$$l(x;\theta) = \sum_{t=0}^{4} log(p_1 N(x^{(t)};\mu_1,\sigma_1^2) + p_2 N(x^{(t)};\mu_2,\sigma_2^2)).$$

   Because of sum of functions inside the log function, it is difficult to maximizing it directly. The EM algorithm provides us an iterative approach to solve for a local optimal solution. Note that the solution is not always a global optimal solution.

   (b) In the first E-step, which examples are more likely to be (but not entirely) assigned to the 2nd Gaussian? In other words, what are the points for which $P(y=2|x^{(i)},\theta_0) > P(y=1|x^{(i)},\theta_0)$?

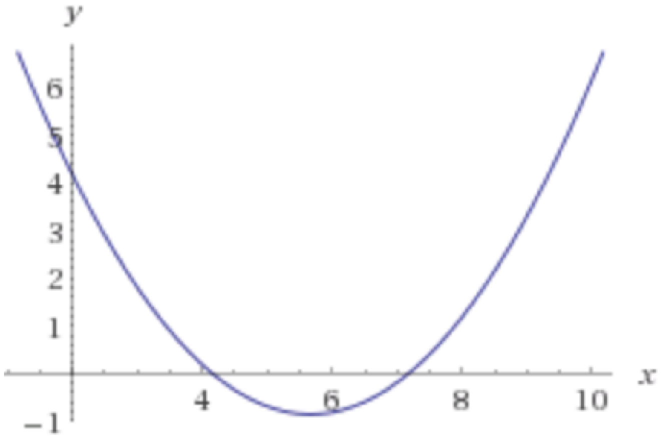
   Note that x will more likely be assigned to Gaussian 2 ($y = 2$) instead of Gaussian 1 ($y = 1$) when the following is true:

$$\begin{aligned}
\frac{P(y=2|x^{(i)},\theta_0)}{P(y=1|x^{(i)},\theta_0)} &> 1 \\
\Leftrightarrow \frac{P(x^{(i)}|y=2)P(y=2)}{P(x^{(i)}|y=1)P(y=1)} &> 1 \\
\Leftrightarrow \frac{\frac{1}{\sqrt{(2\pi\sigma_2^2)}}exp\{-\frac{1}{2}(x-\mu_2)^2/\sigma_2^2\}}{\frac{1}{\sqrt{(2\pi\sigma_1^2)}}exp\{-\frac{1}{2}(x-\mu_1)^2/\sigma_1^2\}} &> 1 \\
\Leftrightarrow \frac{\frac{1}{\sqrt{(2\pi\times 4)}}exp\{-\frac{1}{2}(x-7)^2/4\}}{\frac{1}{\sqrt{(2\pi\times 1)}}exp\{-\frac{1}{2}(x-6)^2\}} &> 1 \\
\Leftrightarrow -\frac{1}{2}exp\{-\frac{1}{2}((x-7)^2/4-(x-6)^2)\} &> 1 \\
\Leftrightarrow \frac{1}{2}exp\{\frac{1}{8}(x-5)(3x-19)\} &> 1 \\
\Leftrightarrow \log(\frac{1}{2}) + \frac{1}{8}(x-5)(3x-19) &> 0 \\
\Leftrightarrow x_1 \approx 4.1525, x_2 \approx 7.1809
\end{aligned}$$

   The plot below is a graph of $\frac{P(y=2|x,\theta_0)}{P(y=1|x,\theta_0)}$. Thus, we can see that all points $x \in [4.15, 7.18]$ have higher probability under class $y = 1$, and all other points have higher probability under $y = 2$. Thus, $x^{(0)}$ and $x^{(1)}$ are more likely (but not entirely) assigned to Gaussian 2, and the rest of the points ($x^{(2)}$, $x^{(3)}$, $x^{(4)}$) are more likely (but not entirely) assigned to Gaussian 1.

   An equivalent way to arrive at the same solution is to simply plug each $x^{(i)}$ into $P(x|y)P(y)$ for each x and each of $y = 1$ and $y = 2$.

Plots:



(x from 1.2 to 10.2)

(c) In the first M-step, in which direction will the two Gaussians move?

Intuitively, Gaussian 2 is influenced most by the points $x^{(0)}$,$x^{(1)}$, and so it will move to the left. Gaussian 1 will be more influenced by the points at $x^{(3)}$, $x^{(4)}$ and $x^{(5)}$ and so it will not move very much to the left. If we computed the actual values, we would see that the updated means for the two Gaussians are approximately $\mu_1 = 5.936$ and $\mu_2 = 2.480$.

(d) In the first M-step, do the variances σ_1^2 and σ_2^2 increase or decrease?

Intuitively, the variance of Gaussian 2 spreads out to cover points $x^{(0)}$ and $x^{(1)}$ which it is most influenced by. The 3 points which most influence Gaussian 1 are concentrated around its mean, we would not expect the variance to increase. Numerically, σ_1^2 decreases to approximately 0.80 while σ_2^2 increases to 2.76.

(e) Where will the two Gaussians be after we run the EM algorithm until convergence? Which of the resulting variances is larger?

Gaussian 1 will be centered around the cluster of 3 points on the right, while Gaussian 2 will be centered around the 2 points on the left. Gaussian 1 will have a larger variance because of the larger size of the right cluster.

Generalization

6. Consider a large set of points uniformly distributed on the unit disk, $x = [x_1, x_2]^T$ such that $|x| <= 1$, and a set of classifiers $h_i(x) = sign(\theta^{(i)}x)$, where $\theta^{(i)} = [cos(\phi_i), sin(\phi_i)]$ with $\phi_i = \frac{i\pi}{100}, i = 1, 2, ..., 200$.

(a) Find the test error for each classifer h_i when $i = 1, 2, ..., 199$.

Assuming that we have sufficient number of data points n uniformly distributed around the unit disk. Since we know that h_{200} correctly classifies all data points, we can infer the label $y = sign(x)$. for any point x. All of the positive points are in the upper half of the axes, and all of the negative points are in the negative half of the axes.

The test error is the probability that a randomly selected point x is misclassified by a classifier $h_i(x)$.

$$\begin{aligned}\mathcal{E}(h) &= 1 - \mathbb{P}\{x \text{ is correctly classified}\}\\ &= 1 - |\frac{i}{100} - 1|\end{aligned}$$

(b) Assume that all the points in the unit disk are all correctly classified by h_{200}. Use the PAC bounds to estimate the number of training points that should be correctly classified by a candidate classifier so that 99.5% of the time, the classifier has a generalization error of less than 0.1.

Assume that we have n training points. Let $\mathcal{H}_\epsilon$ be a set of classifiers for which the generalization error $\mathcal{E}(h) \geq \epsilon$. Let ϵ and δ be the small real numbers that, with probability at least $1 - \delta$ over the choice of training set, any classifier $\hat{h}$ that minimizes the training error has the generalization error $\mathcal{E}(\hat{h}) \leq \epsilon$. We know that there exists a perfect classfier $\hat{h}$ that $\mathcal{E}_n(\hat{h}) = 0$. (There is 0 training error.) Consider a classifier $h \in \mathcal{H}_\epsilon$. The probability that it has 0 training error is $P(E_n(h) = 0) \leq (1 - \epsilon)^n$. There are at most $|H|$ classifiers in $\mathcal{H}_\epsilon$, so in total, we have that the probability of choosing a classfier in $\mathcal{H}_\epsilon$ is $|H| \cdot P(E_n(h) = 0) \leq |H| \cdot (1 - \epsilon)^n \leq |H| \cdot e^{-n\epsilon}$. Then we have that $\delta \leq |H| \cdot e^{-n\epsilon}$ We can rewrite the above as:

$$\begin{aligned}n &= \frac{\log|H| + \log(\frac{1}{\delta})}{\epsilon}\\ &= \frac{\log 200 + \log(\frac{1}{0.005})}{0.1}\\ &\approx 106.\end{aligned}$$