Introduction to Machine Learning, Spring 2025
Homework 6

(Due May 25, 2025 at 11:59pm (CST))

May 6, 2025

1. Please write your solutions in English.
2. Submit your solutions to the course Gradescope.
3. If you want to submit a handwritten version, scan it clearly.
4. Late homeworks submitted within 3 days of the due date will be marked down 25% each day cumulatively Homeworks submitted more than 3 days after the due date will not be accepted unless there is a valid reason, such as a medical or family emergency.
5. You are required to follow ShanghaiTech's academic honesty policies. You are allowed to discuss problems with other students, but you must write up your solutions by yourselves. You are not allowed to copy materials from other students or from online or published resources. Violating academic honesty can result in serious penalties.

1. [25 points] [Boosting]

Suppose that we are interested in learning a classifier, such that at any turn of a game we can pose a question, like "should I attack this ant hill now?", and get an answer.That is, we want to build a classifier which we can feed some features on the current game state, and get the output "attack" or "don't attack". There are many possible ways to define what the action "attack" means, but for now let's define it as sending all friendly ants that can see the ant hill under consideration towards it.

Let's recall the AdaBoost algorithm described in class. Its input is a dataset $\{(x_i, y_i)\}_{i=1}^n$, with x_i being the i-th sample, and $y_i \in \{-1, 1\}$ denoting the i-th label, $i = 1, 2, ..., n$. The features might be composed of a count of the number of friendly ants that can see the ant hill under consideration, and a count of the number of enemy ants these friendly ants can see. For example, if there were 10 friendly ants that could see a particular ant hill, and 5 enemy ants that the friendly ants could see, we would have:

$$x_1 = \begin{bmatrix} 10 \\ 5 \end{bmatrix}$$

The label of the example x_1 is $y_1 = 1$, once the friendly ants were successful in razing the enemy ant hill, and $y_1 = -1$ otherwise. We could generate such examples by running a greedy bot (or any other opponent bot) against a bot that we periodically try to attack an enemy ant hill. Each time this bot tries the attack, we record (say, after 20 turns or some other significant amount of time) whether the attack was successful or not.

(a) Let ϵ_t denote the error of a weak classifier h_t:

$$\epsilon_t = \sum_{i=1}^{n} D_t(i)\mathbb{I}(y_i \neq h_t(x_i))$$

In the simple "attack" / "don't attack" scenario, suppose that we have implemented the following six weak classifiers:

$$h^{(1)}(x_i) = 2 * \mathbb{I}(x_{i1} \geq 2) - 1, \qquad h^{(4)}(x_i) = 2 * \mathbb{I}(x_{i2} \leq 2) - 1,$$
$$h^{(2)}(x_i) = 2 * \mathbb{I}(x_{i1} \geq 6) - 1, \qquad h^{(5)}(x_i) = 2 * \mathbb{I}(x_{i2} \leq 6) - 1,$$
$$h^{(3)}(x_i) = 2 * \mathbb{I}(x_{i1} \geq 10) - 1, \qquad h^{(6)}(x_i) = 2 * \mathbb{I}(x_{i2} \leq 10) - 1.$$

Given ten training data points ($n = 10$) as shown in Table 1:

i	x_{i1}	x_{i2}	y_i
1	1.5	0.5	1
2	2.5	1.5	1
3	3.5	3.5	1
4	6.5	5.5	1
5	7.5	10.5	1
6	1.5	2.5	-1
7	3.5	1.5	-1
8	5.5	5.5	-1
9	7.5	8.5	-1
10	1.5	10.5	-1

Table 1: The training data in (a).

please show that what is the minimum value of ϵ_1 and which of $h^{(1)}, \cdots, h^{(6)}$ achieve this value? Note that there may be multiple classifiers that all have the same ϵ_1. You should list all classifiers that achieve the minimum ϵ_1 value. [5 points]

(b) For all the questions in the remainder of this section, let h_1 denote $h^{(1)}$ chosen in the first round of boosting. (That is, $h^{(1)}$ was the classifier that achieved the minimum ϵ_1.)

(1) What is the value of α_1 (the weight of this first classifier h_1)? [2 points]

(2) What should Z_t be in order to make sure the distribution D_{t+1} is normalized correctly? That is, derive the formula of Z_t in terms of ϵ_t that will ensure $\sum_{i=1}^{n} D_{t+1}(i) = 1$. Please also derive the formula of α_t in terms of ϵ_t. [5 points]

(3) Which points will increase in significance in the second round of boosting? That is, for which points will we have $D_1(i) < D_2(i)$? What are the values of D_2 for these points? [5 points]

(4) In the second round of boosting, the weights on the points will be different, and thus the error ϵ_2 will also be different. Which of $h^{(1)}, \cdots, h^{(6)}$ will minimize ϵ_2? (Which classifier will be selected as the second weak classifier h_2?) What is its value of ϵ_2? [5 points]

(5) What will the average error of the final classifier H be, if we stop after these two rounds of boosting? That is, if $H(x) = \text{sign}(\alpha_1 h_1(x) + \alpha_2 h_2(x))$, what will the training error $\epsilon = \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}(y_i \neq H(x_i))$ be? Is this more, less, or the same as the error we would get, if we just used one of the weak classifiers instead of this final classifier H [3 points]

Solution

(a) Since D_1 is uniform on the training data, so we have $D_1(i) = \frac{1}{10}$ for $i = 1, 2, \cdots, 10$.
So for each classifer $h^{(j)}$, we can get the error $(\epsilon_1)_j$ is

$$(\epsilon_1)_j = \mathbb{E}_{D_1}[\mathbb{I}(y_i \neq h^{(j)}(x_i))] = \sum_{i=1}^{n} D_1(i)\mathbb{I}(y_i \neq h^{(j)}(x_i)) = \frac{1}{10}\sum_{i=1}^{n} \mathbb{I}(y_i \neq h^{(j)}(x_i))$$

- For $h^{(1)}$, we can get that the data x_1, x_7, x_8, x_9 are misclassified, so we have $(\epsilon_1)_1 = \frac{1}{10} \cdot 4 = 0.4$
- For $h^{(2)}$, we can get that the data x_1, x_2, x_3, x_9 are misclassified, so we have $(\epsilon_1)_2 = \frac{1}{10} \cdot 4 = 0.4$
- For $h^{(3)}$, we can get that the data x_1, x_2, x_3, x_4, x_5 are misclassified, so we have $(\epsilon_1)_3 = \frac{1}{10} \cdot 5 = 0.5$
- For $h^{(4)}$, we can get that the data x_3, x_4, x_5, x_7 are misclassified, so we have $(\epsilon_1)_4 = \frac{1}{10} \cdot 4 = 0.4$
- For $h^{(5)}$, we can get that the data x_5, x_6, x_7, x_8 are misclassified, so we have $(\epsilon_1)_5 = \frac{1}{10} \cdot 4 = 0.4$
- For $h^{(6)}$, we can get that the data x_5, x_6, x_7, x_8, x_9 are misclassified, so we have $(\epsilon_1)_6 = \frac{1}{10} \cdot 5 = 0.5$

So above all, the minimum value of ϵ_1 is 0.4, and the classifiers $h^{(1)}, h^{(2)}, h^{(4)}, h^{(5)}$ achieve this value.

(b)(1) From (a), we can get that $\epsilon_1 = 0.4$. So $\alpha_1 = \frac{1}{2}\log\frac{1-\epsilon_1}{\epsilon_1} = \frac{1}{2}\log\frac{1-0.4}{0.4} = \frac{1}{2}\log\frac{3}{2}$.

So above all, $\alpha_1 = \frac{1}{2}\log\frac{3}{2}$.

(2) 1. To make sure the distribution D_{t+1} is normalized correctly, we should make sure $\sum_{i=1}^{n} D_{t+1}(i) = 1$.

Since $D_{t+1}(i) = \frac{D_t(i)}{Z_t}\exp\left(-\alpha_t y_i h_t(x_i)\right)$, so we have

$$\sum_{i=1}^{n} D_{t+1}(i) = \sum_{i=1}^{n} \frac{D_t(i)}{Z_t}\exp\left(-\alpha_t y_i h_t(x_i)\right) = \frac{1}{Z_t}\sum_{i=1}^{n} D_t(i)\exp(-\alpha_t y_i h_t(x_i)) = 1$$

So we have

$$\begin{aligned}
Z_t &= \sum_{i=1}^{n} D_t(i)\exp\left(-\alpha_t y_i h_t(x_i)\right) \\
&= \sum_{i:y_i \neq h_t(x_i)} D_t(i)e^{\alpha_t} + \sum_{i:y_i = h_t(x_i)} D_t(i)e^{-\alpha_t} \\
&= e^{\alpha_t}\sum_{i=1}^{n} D_t(i)\mathbb{I}(y_i \neq h_t(x_i)) + e^{-\alpha_t}\sum_{i=1}^{n} D_t(i)\mathbb{I}(y_i = h_t(x_i)) \\
&= e^{\alpha_t}\epsilon_t + e^{-\alpha_t}(1-\epsilon_t) \qquad \text{(From the definition of } \epsilon_t\text{)}
\end{aligned}$$

2. Then we need to derive α_t in terms of ϵ_t.
Suppose that we have run the AdaBoost algorithm for total T iterations.

Let $H_{\text{final}} = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t\right)$
So we have the final training error is that

$$\begin{aligned}
\epsilon &= \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}(y_i \neq H_{\text{final}}(x_i)) \\
&= \frac{1}{n}\sum_{i=1}^{n} \begin{cases} 1 & \text{if } y_i \neq H_{\text{final}}(x_i) \\ 0 & \text{otherwise} \end{cases} \\
&= \frac{1}{n}\sum_{i=1}^{n} \begin{cases} 1 & \text{if } y_i \left(\sum_{t=1}^{T} \alpha_t h_t\right) \leq 0 \\ 0 & \text{otherwise} \end{cases} \\
&\leq \frac{1}{n}\sum_{i=1}^{n} \exp\left(-y_i\left(\sum_{t=1}^{T} \alpha_t h_t\right)\right)
\end{aligned}$$

Since we totally have T iterations, so for each iteration, we have

$$\begin{aligned}
D_{T+1}(i) &= \frac{D_T(i)}{Z_T}\exp(-\alpha_T y_i h_T(x_i)) \\
D_T(i) &= \frac{D_{T-1}(i)}{Z_{T-1}}\exp(-\alpha_{T-1} y_i h_{T-1}(x_i)) \\
&\vdots \\
D_2(i) &= \frac{D_1(i)}{Z_1}\exp(-\alpha_1 y_i h_1(x_i)) \\
D_1(i) &= \frac{1}{n}
\end{aligned}$$

Multiply these equations, we can get that

$$D_{T+1}(i) = \frac{1}{n}\prod_{t=1}^{T}\frac{1}{Z_t}\exp(-\alpha_t y_i h_t(x_i)) = \frac{1}{n}\cdot\frac{1}{\prod_{t=1}^{T} Z_t}\cdot\exp\left(-y_i\sum_{t=1}^{T}\alpha_t h_t(x_i)\right)$$

i.e.

$$\frac{1}{n}\cdot\exp\left(-y_i\sum_{t=1}^{T}\alpha_t h_t(x_i)\right) = D_{T+1}(i)\prod_{t=1}^{T} Z_t$$

If we put this into the final training error, we can get that

$$\epsilon \leq \frac{1}{n}\sum_{i=1}^{n}\exp\left(-y_i\left(\sum_{t=1}^{T}\alpha_t h_t\right)\right) = \sum_{i=1}^{n} D_{T+1}(i)\prod_{t=1}^{T} Z_t = \prod_{t=1}^{T} Z_t\left(\sum_{i=1}^{n} D_{T+1}(i)\right)$$

Since Z_t is to make sure D_{t+1} is normalized correctly, so we have $\sum_{i=1}^{n} D_{T+1}(i) = 1$, so we have

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

So if we want to minimize the final error ϵ, we should minimize $\prod_{t=1}^{T} Z_t$. i.e. we should minimize Z_t for each $t = 1, 2, \cdots, T$.
So for each $Z_t = e^{\alpha_t}\epsilon_t + e^{-\alpha_t}(1-\epsilon_t)$:

$$\begin{aligned}
\frac{\partial Z_t}{\partial \alpha_t} &= \epsilon_t e^{\alpha_t} - (1-\epsilon_t)e^{-\alpha_t} \\
\frac{\partial^2 Z_t}{\partial \alpha_t^2} &= \epsilon_t e^{\alpha_t} + (1-\epsilon_t)e^{-\alpha_t} > 0
\end{aligned}$$

So we can get that Z_t is a convex function of α_t.
So to minimize Z_t, we should make $\dfrac{\partial Z_t}{\partial \alpha_t} = 0$.
i.e.

$$\begin{aligned} \epsilon_t e^{\alpha_t} &= (1-\epsilon_t)e^{-\alpha_t} \\ e^{2\alpha_t} &= \frac{1-\epsilon_t}{\epsilon_t} \\ \alpha_t &= \frac{1}{2}\log\frac{1-\epsilon_t}{\epsilon_t} \end{aligned}$$

Since $\epsilon_t = \mathbb{E}_{D_t}[\mathbb{I}(y_i \neq h_t(x_i))] = P_{D_t}(y_i \neq h_t(x_i))$, so $\epsilon_t \in (0,1)$.
So we have $\dfrac{1-\epsilon_t}{\epsilon_t} > 0$, so $\log\dfrac{1-\epsilon_t}{\epsilon_t}$ is valid.
So we have derived that $\alpha_t = \dfrac{1}{2}\log\dfrac{1-\epsilon_t}{\epsilon_t}$.
And put it into the equation (7), we can get that

$$Z_t = \epsilon_t e^{\alpha_t} + (1-\epsilon_t)e^{-\alpha_t} = \epsilon_t\sqrt{\frac{1-\epsilon_t}{\epsilon_t}} + (1-\epsilon_t)\sqrt{\frac{\epsilon_t}{1-\epsilon_t}} = 2\sqrt{\epsilon_t(1-\epsilon_t)}$$

So above all, we have derived that

$$\begin{aligned} Z_t &= 2\sqrt{\epsilon_t(1-\epsilon_t)} \\ \alpha_t &= \frac{1}{2}\log\frac{1-\epsilon_t}{\epsilon_t} \end{aligned}$$

(3) From (2), we can get that $Z_1 = 2\sqrt{\epsilon_1(1-\epsilon_1)} = 2\sqrt{0.4\cdot 0.6} = 0.4\sqrt{6}$.
And $\alpha_1 = \dfrac{1}{2}\log\dfrac{1-\epsilon_1}{\epsilon_1} = \dfrac{1}{2}\log\dfrac{1-0.4}{0.4} = \dfrac{1}{2}\log\dfrac{3}{2}$.
Since we take $h_1 = h^{(1)}$, so

$$D_2(i) = \frac{D_1(i)}{Z_1}\exp\left(-\alpha_1 y_i h_1(x_i)\right) = \frac{1}{10\cdot 0.4\sqrt{6}}\exp\left(-\frac{1}{2}\log\frac{3}{2}\cdot y_i\cdot h^{(1)}(x_i)\right)$$

From (a), we can get that for points x_1, x_7, x_8, x_9, which are misclassified, so we have $y_i \cdot h^{(1)}(x_i) = -1$.
So their weight $D_2(i) = \dfrac{1}{10\cdot 0.4\sqrt{6}}\exp\left(-\dfrac{1}{2}\log\dfrac{3}{2}\cdot(-1)\right) = \dfrac{1}{10\cdot 0.4\sqrt{6}}\cdot\sqrt{\dfrac{3}{2}} = \dfrac{1}{8} > D_1(i) = \dfrac{1}{10}$.
And for other points $x_2, x_3, x_4, x_5, x_6, x_{10}$, which are correctly classified, so we have $y_i \cdot h^{(1)}(x_i) = 1$.
So their weight $D_2(i) = \dfrac{1}{10\cdot 0.4\sqrt{6}}\exp\left(-\dfrac{1}{2}\log\dfrac{3}{2}\cdot 1\right) = \dfrac{1}{10\cdot 0.4\sqrt{6}}\cdot\sqrt{\dfrac{2}{3}} = \dfrac{1}{12} < D_1(i) = \dfrac{1}{10}$.

So above all, the misclassified points x_1, x_7, x_8, x_9 will increase in significance in the second round of boosting, and their weight $D_2(i) = \dfrac{1}{8}$.

(4) From (3), we know that $D_2(1) = D_2(7) = D_2(8) = D_2(9) = \dfrac{1}{8}$,
and $D_2(2) = D_2(3) = D_2(4) = D_2(5) = D_2(6) = D_2(10) = \dfrac{1}{12}$.
So for each classifer $h^{(j)}$, we can get the error $(\epsilon_2)_j$ is

$$(\epsilon_2)_j = \mathbb{E}_{D_2}[\mathbb{I}(y_i \neq h^{(j)}(x_i))] = \sum_{i=1}^{n} D_2(i)\mathbb{I}(y_i \neq h^{(j)}(x_i))$$

- For $h^{(1)}$, we have $(\epsilon_2)_1 = \dfrac{1}{8}\cdot 4 + \dfrac{1}{12}\cdot 0 = \dfrac{1}{2}$.
- For $h^{(2)}$, we have $(\epsilon_2)_2 = \dfrac{1}{8}\cdot 2 + \dfrac{1}{12}\cdot 2 = \dfrac{5}{12}$.
- For $h^{(3)}$, we have $(\epsilon_2)_3 = \dfrac{1}{8}\cdot 1 + \dfrac{1}{12}\cdot 4 = \dfrac{11}{24}$.
- For $h^{(4)}$, we have $(\epsilon_2)_4 = \dfrac{1}{8}\cdot 1 + \dfrac{1}{12}\cdot 3 = \dfrac{3}{8}$.

- For $h^{(5)}$, we have $(\epsilon_2)_5 = \frac{1}{8} \cdot 2 + \frac{1}{12} \cdot 2 = \frac{5}{12}$.
- For $h^{(6)}$, we have $(\epsilon_2)_6 = \frac{1}{8} \cdot 3 + \frac{1}{12} \cdot 2 = \frac{13}{24}$.

So above all, the minimum value of ϵ_2 is $\frac{3}{8}$, and the classifier $h^{(4)}$ achieve this value.

(5) From (1), we can get that $\alpha_1 = \frac{1}{2} \log \frac{3}{2}$.
And from (4), we can get that $\epsilon_2 = \frac{3}{8}$.
So $\alpha_2 = \frac{1}{2} \log \frac{1-\epsilon_2}{\epsilon_2} = \frac{1}{2} \log \frac{1-\frac{3}{8}}{\frac{3}{8}} = \frac{1}{2} \log \frac{5}{3}$.
And since $h_1(x) = h^{(1)}(x)$ and $h_2(x) = h^{(4)}(x)$, so

$$H(x) = \text{sign}(\alpha_1 h_1(x) + \alpha_2 h_2(x)) = \text{sign}\left(\frac{1}{2} \log \frac{3}{2} h^{(1)}(x) + \frac{1}{2} \log \frac{5}{3} h^{(4)}(x)\right)$$

There are total 4 possible combinations of $h^{(1)}(x)$ and $h^{(4)}(x)$, which are $(-1,-1), (1,-1), (-1,1), (1,1)$. So we can get that

- For $(-1,-1)$, we have $H(x) = \text{sign}\left(\frac{1}{2} \log \frac{3}{2} \cdot (-1) + \frac{1}{2} \log \frac{5}{3} \cdot (-1)\right) = -1$.
- For $(1,-1)$, we have $H(x) = \text{sign}\left(\frac{1}{2} \log \frac{3}{2} \cdot 1 + \frac{1}{2} \log \frac{5}{3} \cdot (-1)\right) = \text{sign}(\frac{1}{2} \log \frac{9}{10}) = -1$.
- For $(-1,1)$, we have $H(x) = \text{sign}\left(\frac{1}{2} \log \frac{3}{2} \cdot (-1) + \frac{1}{2} \log \frac{5}{3} \cdot 1\right) = \text{sign}(\frac{1}{2} \log \frac{10}{9}) = 1$.
- For $(1,1)$, we have $H(x) = \text{sign}\left(\frac{1}{2} \log \frac{3}{2} \cdot 1 + \frac{1}{2} \log \frac{5}{3} \cdot 1\right) = 1$.

So we can get that x_3, x_4, x_5, x_7 are misclassified by $H(x)$(actually, $H(x)$ is exactly same with $h^{(2)}(x)$), so we have

$$\epsilon = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}(y_i \neq H(x_i)) = \frac{1}{10} \cdot 4 = 0.4$$

And we have $\epsilon_1 = \min_{i=1,2,\cdots,6} (\epsilon_1)_i = \min\{0.4, 0.4, 0.5, 0.4, 0.4, 0.5\} = 0.4$, so we can get that $\epsilon = \epsilon_1$.

So above all, the average error of the final classifier H is 0.4, and it is the same as the error we would get, if we just used one of the weak classifiers instead of this final classifier H.

2. [10 points] [Equivalence of PCA objectives]

Consider a dataset of n observations $\mathbf{X} \in \mathbb{R}^{n \times d}$, and our goal is to project the data onto a subspace having dimensionality p, $p < d$. Prove that PCA based on projected variance maximization is equivalent to PCA based on projected error (Euclidean error) minimization.

Solution

Suppose that all sampled points are centered, so the sample mean is $\mu = \mathbf{0}$.
And suppose that $\mathbf{v}$ is the direction of the projection. Where $\mathbf{v} \in \mathbb{R}^d$ and let $\|\mathbf{v}\| = 1$.
So for each sampled point X_i, the projection of X_i on the direction $\mathbf{v}$ is $X_i \cdot \mathbf{v} = X_i^\top \mathbf{v}$.
And for the PCA problem, our goal is to find the most suitable p directions $\mathbf{v}$. We could consider them seperately, and with the method to take the most p suitable directions $\mathbf{v}$.

1. The method based on projected variance maximization:

The mean of the projection values is that $\mu' = \frac{1}{n}\sum_{i=1}^{n}\left(X_i^\top \mathbf{v}\right) = \mathbf{v}^\top(\frac{1}{n}\sum_{i=1}^{n} X_i) = \mathbf{v}^\top \mu = \mathbf{0}$, since $\mu = \mathbf{0}$.
So the objective function is to maximize the projected variance, which is

$$\max_{\mathbf{v}} \frac{1}{n}\sum_{i=1}^{n}\left(X_i^\top \mathbf{v}\right)^2$$

2. As for the method based on projected error minimization:
The objective function is to minimize the projected error, which is

$$\min_{\mathbf{v}} \sum_{i=1}^{n} \|X_i - \left(X_i^\top \mathbf{v}\right)\mathbf{v}\|^2$$

From the vector's addition operation, we can get that $X_i - \left(X_i^\top \mathbf{v}\right)\mathbf{v}$ is perpendicular to $\mathbf{v}$.
So we have $(X_i - \left(X_i^\top \mathbf{v}\right)\mathbf{v}) \cdot \mathbf{v} = 0$. So

$$\|X_i\|^2 = \|\left(X_i - \left(X_i^\top \mathbf{v}\right)\mathbf{v}\right) + \left(\left(X_i^\top \mathbf{v}\right)\mathbf{v}\right)\|^2 = \left\|X_i - \left(X_i^\top \mathbf{v}\right)\mathbf{v}\right\|^2 + \left\|\left(X_i^\top \mathbf{v}\right)\mathbf{v}\right\|^2$$

Since $\|\mathbf{v}\| = 1$, so

$$\begin{aligned} & \|\left(X_i^\top \mathbf{v}\right)\mathbf{v}\|^2 = \left(X_i^\top \mathbf{v}\right)^2 \|\mathbf{v}\|^2 = \left(X_i^\top \mathbf{v}\right)^2 \\ \Rightarrow \quad & \|X_i - \left(X_i^\top \mathbf{v}\right)\mathbf{v}\|^2 = \|X_i\|^2 - \left(X_i^\top \mathbf{v}\right)^2 \end{aligned}$$

So the objective function is equivalent to

$$\min_{\mathbf{v}} \sum_{i=1}^{n} \|X_i - \left(X_i^\top \mathbf{v}\right)\mathbf{v}\|^2 = \min_{\mathbf{v}} \sum_{i=1}^{n} \|X_i\|^2 - \sum_{i=1}^{n}\left(X_i^\top \mathbf{v}\right)^2$$

Since our goal is to find the suitable $\mathbf{v}$, so the sample points X_i is fixed.
So $\sum_{i=1}^{n} \|X_i\|^2$ is a constant. And n is also a constant.
So the objective function is equivalent to

$$\min_{\mathbf{v}} \sum_{i=1}^{n} \|X_i\|^2 - \sum_{i=1}^{n}\left(X_i^\top \mathbf{v}\right)^2 \Leftrightarrow \min_{\mathbf{v}} -\sum_{i=1}^{n}\left(X_i^\top \mathbf{v}\right)^2 \Leftrightarrow \max_{\mathbf{v}} \sum_{i=1}^{n}\left(X_i^\top \mathbf{v}\right)^2 \Leftrightarrow \max_{\mathbf{v}} \frac{1}{n}\sum_{i=1}^{n}\left(X_i^\top \mathbf{v}\right)^2$$

So above all, the objective function of the method based on projected error minimization is the same as the objective function of the method based on projected variance maximization.
And they also have the same constrain that is $\|\mathbf{v}\| = 1$.
So the two method is actually the same optimization problem.
So PCA based on projected variance maximization is equivalent to PCA based on projected error minimization.

3. [15 points] [Performing PCA by Hand]

Let's do principal components analysis (PCA)! Consider this sample of six points $X_i \in \mathbb{R}^2$.

$$\left\{ \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \end{bmatrix} \right\}$$

(a) Compute the mean of the sample points and write the centered design matrix $\dot{X}$. [4 points] (Hint: The sample mean is by subtracting the mean from each sample.)

(b) Find all the principal components of this sample. Write them as unit vectors. [5 points] (Hint: The principal components of our dataset are the eigenvectors of the matrix $\dot{X}^\top \dot{X} =$. The characteristic polynomial of this symmetric matrix is $\det\left(\lambda I - \dot{X}^\top \dot{X}\right)$.)

(c) Which of those two principal components would be preferred if you use only one? [2 points]
What information does the PCA algorithm use to decide that one principal components is better than another? [2 points]
From an optimization point of view, why do we prefer that one? [2 points]

Solution

(a) Original sample matrix $X \in \mathbb{R}^{n \times d} = \mathbb{R}^{6 \times 2}$.

The sample mean is that $\mu = \frac{1}{6} \sum_{i=1}^{6} X_i = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$. After subtracting the mean from each sample, we form the centered design matrix

$$\dot{X} = X - \mu = \begin{bmatrix} -1 & -1 \\ -1 & 0 \\ 0 & -1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix}$$

(b) We can calculate that

$$\dot{X}^\top \dot{X} = \begin{bmatrix} 4 & 2 \\ 2 & 4 \end{bmatrix}$$

The characteristic polynomial of this symmetric matrix is

$$\det(\lambda I - \dot{X}^\top \dot{X}) = (\lambda - 2)(\lambda - 6)$$

So the eigenvalues of $\dot{X}^\top \dot{X}$ are $\lambda_1 = 6, \lambda_2 = 2$.

For $\lambda_1 = 6$, we have the corresponding eigenvector is that $\mathbf{v}_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$.

And for $\lambda_2 = 2$, we have the corresponding eigenvector is that $\mathbf{v}_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}$.

So above all, the principal components of this sample are $\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ and $\frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}$.

(c) 1. Since $\lambda_1 = 6 > \lambda_2 = 2$, so we prefer $\mathbf{v}_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ if we use only one principal component.

2. The PCA algorithm use the variance of the data projected onto the corresponding eigenvector $\mathbf{v}$ or the minimum projected error to decide that one principal components is better than another.
Or we can say that the PCA algorithm use the eigenvalue of the matrix $\dot{X}^\top \dot{X}$ to decide that one principal components is better than another.

3. From an optimization point of view, we prefer $\mathbf{v}_1$ because the variance of the data projected onto $\mathbf{v}_1$ is larger than the variance of the data projected onto $\mathbf{v}_2$. And since λ is the eigenvalue of $\dot{X}^\top \dot{X}$, so

$$\begin{aligned} \dot{X}^\top \dot{X}\mathbf{v} &= \lambda \mathbf{v} \\ \mathbf{v}^\top \dot{X}^\top \dot{X}\mathbf{v} &= \mathbf{v}^\top \lambda \mathbf{v} && \text{(multiply } \mathbf{v}^\top \text{to the left on both sides)} \\ \mathbf{v}^\top \dot{X}^\top \dot{X}\mathbf{v} &= \lambda && (\mathbf{v}^\top \mathbf{v} = \|\mathbf{v}\|^2 = 1) \end{aligned}$$

Also, the variance of the data projected onto $\mathbf{v}$ is

$$\begin{aligned}
\dot{\sigma}^2 &= \frac{1}{n}\sum_{i=1}^{n}\left(\dot{X_i}^\top \mathbf{v}\right)^2 \qquad \text{(the centered designed } \dot{X}_i \text{ is with mean 0)} \\
&= \frac{1}{n}\sum_{i=1}^{n}\mathbf{v}^\top \dot{X}_i \dot{X_i}^\top \mathbf{v} \\
&= \mathbf{v}^\top \left(\frac{1}{n}\sum_{i=1}^{n}\dot{X}_i \dot{X_i}^\top\right)\mathbf{v} \\
&= \mathbf{v}^\top \left(\frac{1}{n}\dot{X}^\top \dot{X}\right)\mathbf{v} \qquad \text{(the covirance matrix of the centered designed } \dot{X} \text{ is } \frac{1}{n}\dot{X}^\top\dot{X}) \\
&= \frac{1}{n}\mathbf{v}^\top \dot{X}^\top \dot{X}\mathbf{v}
\end{aligned}$$

So $\lambda = n\dot{\sigma}^2$.

Since the sample points' number n is a constant, so we can use the eigenvalue to represent the variance of the data projected onto the corresponding eigenvector.