STANFORD UNIVERSITY
CS 229, Spring 2016
Midterm Examination

Monday, May 9, 6:00pm-9:00pm

Question	Points
1 Short answers	/21
2 Exponential families	/7
3 Local polynomial regression	/15
4 Not stochastic gradient descent	/14
5 Randomized kernels	/17
6 Linear regression boosting	/30
Total	/104

Name of Student: _____

SUNetID: _____@stanford.edu

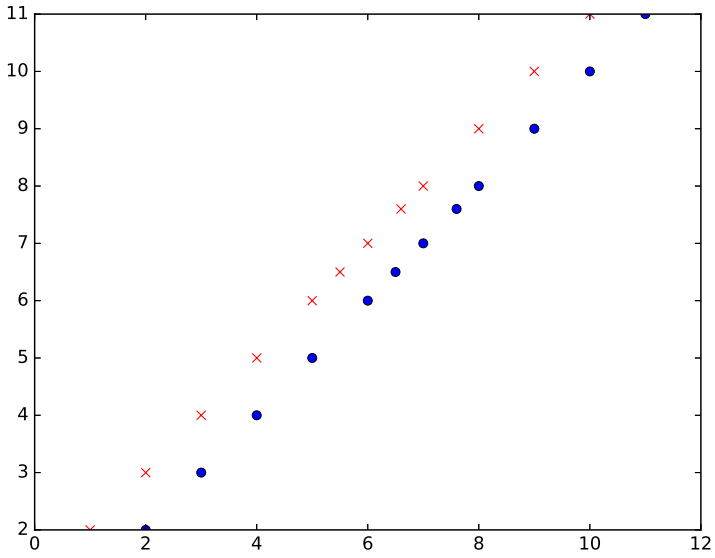
The Stanford University Honor Code:

I attest that I have not given or received aid in this examination, and that I have done my share and taken an active part in seeing to it that others as well as myself uphold the spirit and letter of the Honor Code.

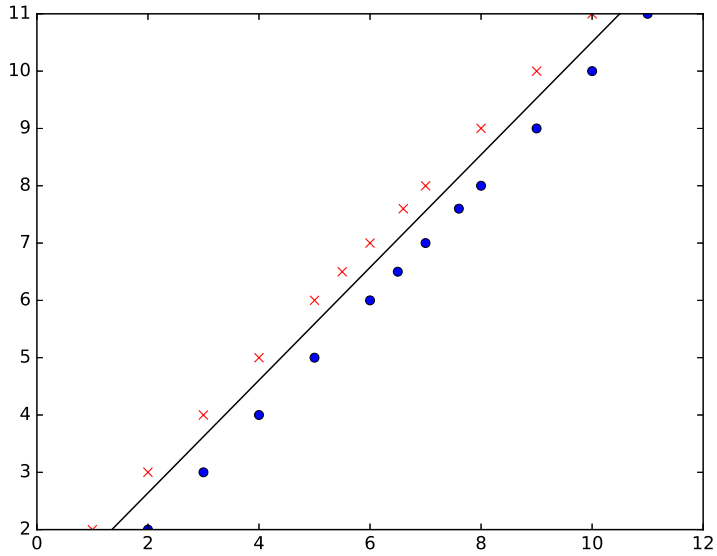
Signed: _____

1. **[21 points] Short Answer**

   (a) [6 points] For the given data plots below, choose a method you could use to classify the data, and a method that is not reasonable to use for the given dataset. Draw (an approximation to) the regions the classifier would classify as positive and negative and briefly explain the performance of two methods per graph.

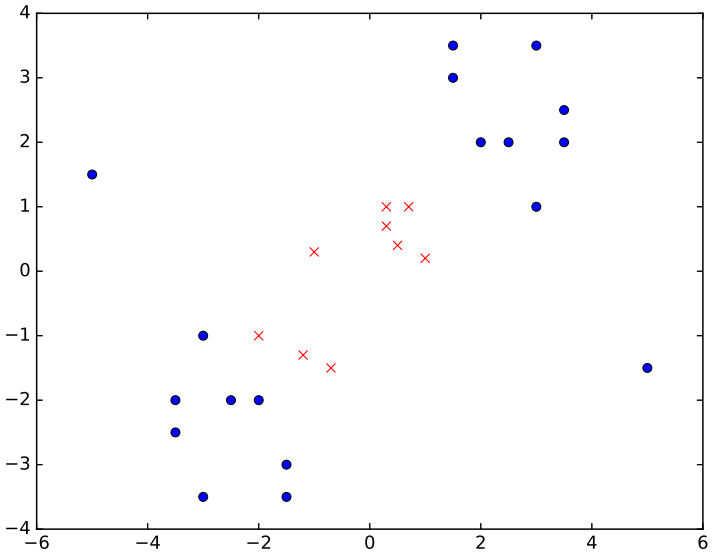


   A supervised learning method that would likely work:
   Answer:
   Support Vector Machine - The data points are linearly separable along a diagonal

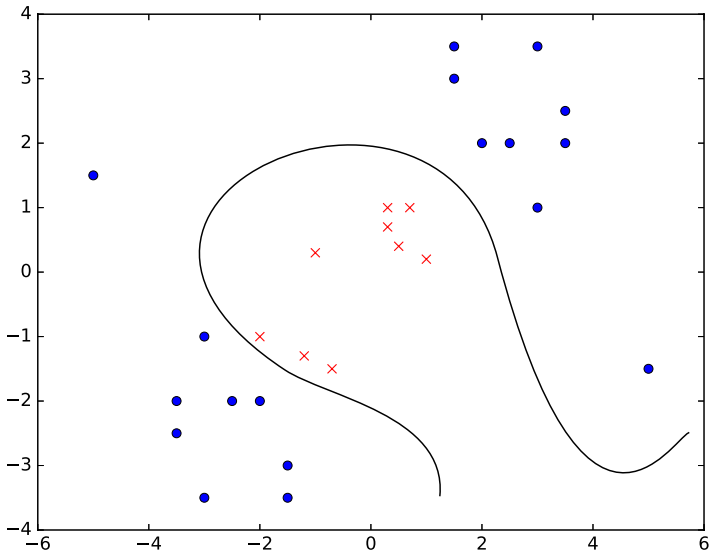


   line.

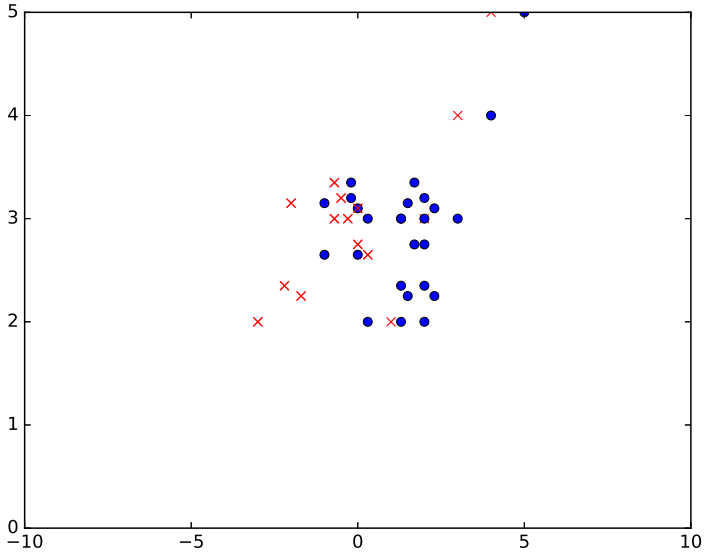
A supervised learning method that would likely not work:
Answer: Naive Bayes - assumption of independence between features is not helpful when the data is strongly correlated.



A supervised learning method that would likely work:
Answer: Kernel perceptron - Can separate around the curve of the graph using the kernel trick.



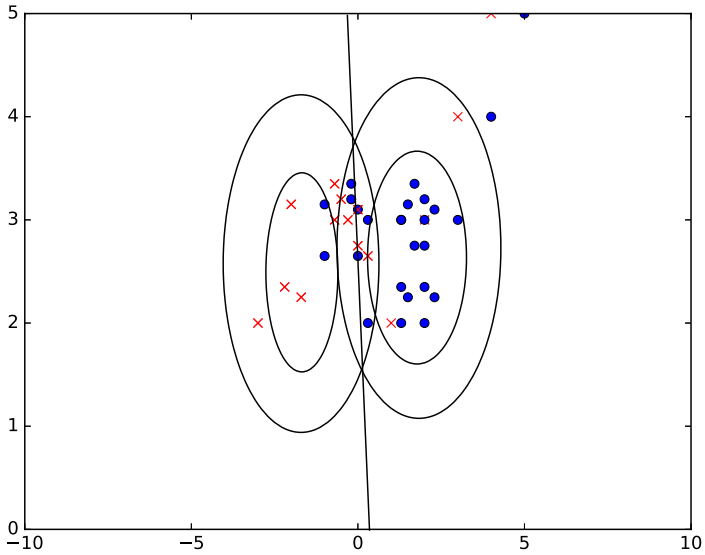
A supervised learning method that would likely not work:
Answer: Linear classification - Data is not linearly separable.



A supervised learning method that would likely work:

Answer:

Gaussian Discriminant Analysis - Has room for error, identifies main two clusters with overlap.



A supervised learning method that would likely not work:

Answer: We accepted essentially anything reasonable here. A method that attempts to classify every example perfectly might fail, for example, because the data clusters overlap.

(b) [4 points] We attempt to separate the dataset in Figure 1 (positive labels are `x`'s

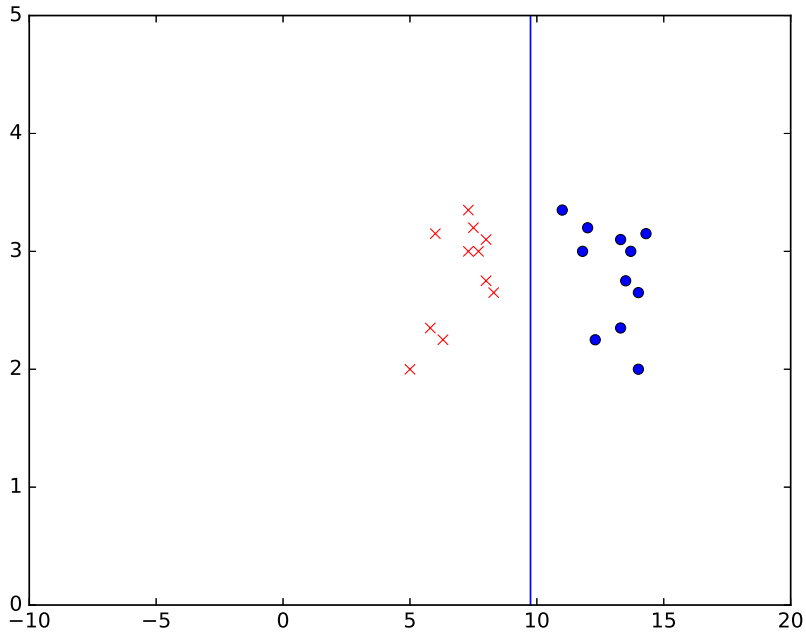


Figure 1: An easy to separate dataset.

and negative are o's) using the loss functions

$$\mathsf{L}_1(\theta^T x, y) = \frac{1}{2}(\theta^T x - y)^2$$
$$\mathsf{L}_2(\theta^T x, y) = \left[1 - y\theta^T x\right]_+ = \max\{0, 1 - y\theta^T x\}.$$

For the given dataset, we plot the line $\{x \in \mathbb{R}^2 : x^T\theta^* = 0\}$, where θ^* is the minimizer of the average losses (empirical risks) $J_1(\theta) = \frac{1}{m}\sum_{i=1}^m \mathsf{L}_1(\theta^T x, y)$ and $J_2(\theta) = \frac{1}{m}\sum_{i=1}^m \mathsf{L}_2(\theta^T x, y)$. (For the given dataset, the same θ is optimal for each.)

i. [1 points] What are the names of the loss functions?
Answer: L_1 is the least-squares loss or squared error, L_2 is the hinge loss (or SVM loss).

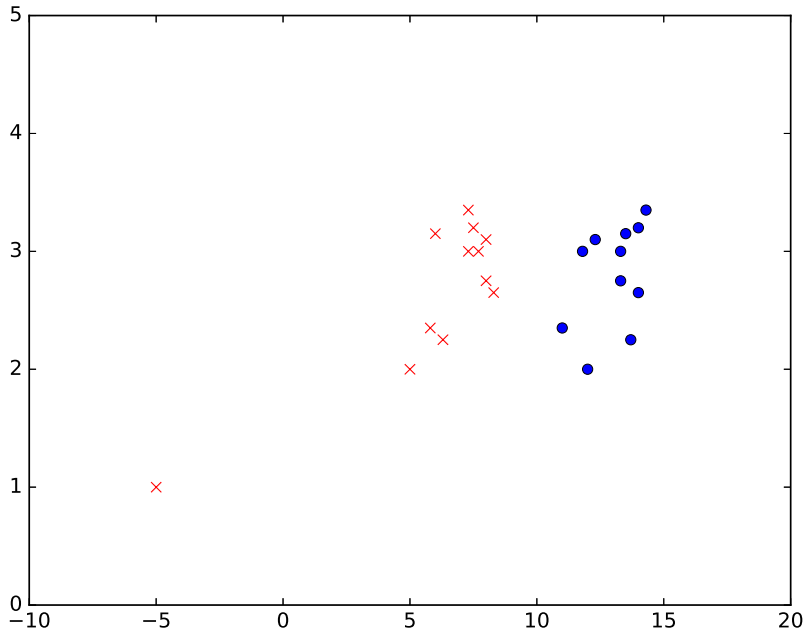
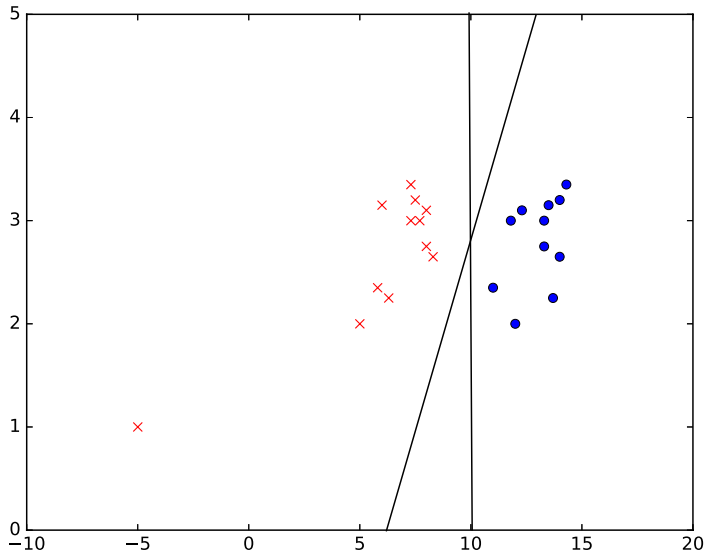


Figure 2: A new point.

We add new data point with positive label at the point (-5, 1), as in Fig. 2.

ii. [3 points] Could the classifying line change for either of the loss functions? Briefly explain why. Draw (your best estimate of) the new classification boundaries, and clearly label the lines with the corresponding loss functions.
Answer: For the hinge loss, nothing changes—the point is classified perfectly and suffers no loss. For the squared error, the loss changes substantially, because we try to assign it a prediction *close* to 1.



(c) [2 points] Suppose we have two collections of hypotheses, H_1 and H_2, and we fit them on a training set to give $\widehat{h}_1$ and $\widehat{h}_2$ solving

$$\widehat{h}_1 = \operatorname*{argmin}_{h \in H_1} \frac{1}{m} \sum_{i=1}^{m} \mathbf{1}\left\{h(x^{(i)}) \neq y^{(i)}\right\} \text{ and } \widehat{h}_2 = \operatorname*{argmin}_{h \in H_2} \frac{1}{m} \sum_{i=1}^{m} \mathbf{1}\left\{h(x^{(i)}) \neq y^{(i)}\right\}.$$

We have $\mathrm{VC}(H_1) < \mathrm{VC}(H_2)$. Which of $\widehat{h}_1$ and $\widehat{h}_2$ will have lower training error?

Answer: It depends. Neither will necessarily be lower, because the hypothesis classes could be completely unrelated.

(d) [3 points] Give an example of a class of hypotheses H and a distribution on (x, y), where $x \in \mathbb{R}$ and $y \in \{-1, 1\}$, such that there always exists $h \in H$ with

$$\frac{1}{m} \sum_{i=1}^{m} \mathbf{1}\left\{h(x^{(i)}) \neq y^{(i)}\right\} < .01 \text{ and } P(h(X) \neq Y) > .99$$

no matter the training set size m.

Answer: Here is one example; many others are possible. If we take H to be the class of all functions $h : \mathbb{R} \to \{-1, 1\}$, and then let P be a distribution with $x \sim \mathcal{N}(0, 1)$ and $Y = \operatorname{sign}(x)$. We can always find a function $h \in H$ perfectly classifying the training data but for all $x \notin \{x^{(1)}, \ldots, x^{(m)}\}$ predicting $h(x) = -\operatorname{sign}(x)$. Thus $P(h(X) \neq Y) = 1$ but the empirical error is always 0.

(e) [3 points] You are given the choice of two loss functions for a binary classification problem: the exponential and logistic losses,

$$\mathsf{L}(\theta^T x, y) = \exp(-y\theta^T x) \text{ or } \mathsf{L}(\theta^T x, y) = \log(1 + \exp(-y\theta^T x)).$$

The label $y^{(i)}$ is incorrect for about 10% (the precise number is unimportant) of the training data $\{(x^{(i)}, y^{(i)})\}_{i=1}^{m}$. You will choose a hypothesis by minimizing

$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathsf{L}(\theta^T x^{(i)}, y^{(i)})$ for one of the two losses. Which loss is more likely to have better generalization performance? Justify your answer.

Answer: The logistic loss. It is less sensitive to mistaken labels, as it grows only linearly for mis-classifications rather than exponentially. So the exponential loss will work very hard to classify mis-labeled examples correctly.

(f) [3 points] Instead of minimizing the average loss on a training set, John decides to minimize the maximal loss on your training set for a classification problem with $y \in \{-1, 1\}$. He has training data $x^{(i)} \in \mathbb{R}$, and he will learn a linear classifier $\theta x^{(i)}$ by finding $\theta \in \mathbb{R}$. He decides to minimize

$$J_{\max}(\theta) = \max_{i \in \{1,\ldots,m\}} \log(1 + \exp(-y^{(i)} x^{(i)} \theta)).$$

Examples 1 and 2 in his dataset satisfy $x^{(1)} < 0$ and $x^{(2)} < 0$, but $y^{(1)} \neq y^{(2)}$. Is his idea to minimize the maximal loss a good one? Why or why not? [*Hint:* The answer is not "It depends."]

Answer: No, it's a bad idea. The minimizing θ will be $\theta = 0$, as this is the only way to guarantee that

$$\max_i \log(1 + \exp(-y^{(i)} x^{(i)} \theta)) \leq \log 2,$$

which is attained at $\theta = 0$.

2. **[7 points] Exponential families and generative models**

We have a problem with k categories, $y \in \{1, \ldots, k\}$, and we make the generative assumption that x conditional on y follows the exponential family distribution

$$p(x \mid y; \eta) = b(x) \exp\left(\eta_y^T T(x) - A(\eta; y)\right)$$

where $\eta_y \in \mathbb{R}^n$ for $y = 1, \ldots, k$. Also assume that we have prior probabilities

$$p(y) = \pi_y > 0 \quad \text{for } y = 1, \ldots, k.$$

Show that the distribution of y conditional on x follows the multinomial logistic model. That is, show that there are $\theta_y \in \mathbb{R}^n$ (for $y = 1, \ldots, k$) and $\theta^{(0)} \in \mathbb{R}^k$ such that

$$p(y \mid x) = \frac{\exp\left(\theta_y^{(0)} + \theta_y^T T(x)\right)}{\sum_{l=1}^k \exp\left(\theta_l^{(0)} + \theta_l^T T(x)\right)}.$$

Describe explicitly what the values of θ and $\theta^{(0)}$ are as a function of η, π, and A.

Answer: We use Bayes' rule, which gives

$$\begin{aligned} p(y \mid x) &= \frac{p(x \mid y)p(y)}{\sum_{l=1}^k p(x \mid l)p(l)} = \frac{b(x) \exp(\eta_y^T T(x) - A(\eta; y))\pi_y}{\sum_{l=1}^k b(x) \exp(\eta_l^T T(x) - A(\eta; l))\pi_l} \\ &= \frac{\exp(\eta_y^T T(x) - A(\eta; y) + \log \pi_y)}{\sum_{l=1}^k \exp(\eta_l^T T(x) - A(\eta; l) + \log \pi_l)}. \end{aligned}$$

Now, let $\theta_l = \eta_l$ and $\theta_l^{(0)} = -A(\eta; l) + \log \pi_l$, which gives

$$p(y \mid x) = \frac{\exp\left(\theta_y^T T(x) + \theta_y^{(0)}\right)}{\sum_{l=1}^k \exp(\theta_l^T T(x) + \theta_l^{(0)})}$$

as desired.

3. **[15 points] Local Polynomial Regression**

We have a training set:

$$S = \{(x^{(i)}, y^{(i)}), i = 1, ..., m\} \text{ where } x^{(i)} \in \mathbb{R}^n, y^{(i)} \in \mathbb{R}.$$

Assume $x^{(i)}$ contains the intercept term (i.e. $x_0^{(i)} = 1$ for all i). Consider the following regression model:

$$y = {\theta^{(1)}}^T x + {\theta^{(2)}}^T x^2 + \cdots + {\theta^{(p-1)}}^T x^{p-1} + {\theta^{(p)}}^T x^p$$

where $\theta^{(p)}$ denotes the p^{th} parameter vector and where x^p denotes element-wise exponentiation (i.e. each element of x is raised to the p^{th} power). The cost function for this model is:

$$J(\theta^{(1)}, \ldots, \theta^{(p)}) = \frac{1}{2} \sum_{i=1}^{m} w^{(i)} \left(\sum_{k=1}^{p} {\theta^{(k)}}^T {x^{(i)}}^k - y^{(i)} \right)^2.$$

As before, $w^{(i)}$ is the "weight" for a specific training example i.

(a) [3 points] Show that $J(\theta^{(1)}, ..., \theta^{(p)})$ can be written as:

$$J(\theta) = \frac{1}{2} \text{tr} \Big[(X\theta - y)^T W (X\theta - y) \Big].$$

Using $\theta^{(1)}, ..., \theta^{(p)}$, you need to define a vector θ and matrices X and W such that the transformation is possible. Clearly state the dimensions of these variables.

Answer: Define θ to be the concatenation of each $\theta^{(1)}, ..., \theta^{(p)}$, that is:

$$\theta = (\theta_1^{(1)}, ..., \theta_n^{(1)}, \theta_1^{(2)}, ..., \theta_n^{(2)}, ..., ..., \theta_1^{(p-1)}, ..., \theta_n^{(p-1)}, \theta_1^{(p)}, ..., \theta_n^{(p)})$$

Similarly, we redefine $x^{(i)}$ such that:

$$x^{(i)} = ({x_1^{(i)}}^1, ..., {x_n^{(i)}}^1, {x_1^{(i)}}^2, ..., {x_n^{(i)}}^2, ..., ..., {x_1^{(i)}}^{p-1}, ..., {x_n^{(i)}}^{p-1}, {x_1^{(i)}}^p, ..., {x_n^{(i)}}^p)$$

We define W to be a diagonal matrix with $W_{ii} = w^{(i)}$, similar to the problem set. The dimensions are:

- θ is a vector of size pn
- X is the design matrix of size $m \times pn$
- W is a matrix of size $m \times m$

Some students included the bias term such that $x^{(i)} \in \mathbb{R}^{n+1}$. This resulted in the dimensions:

$$\theta \in \mathbb{R}^{pn+p} \qquad X \in \mathbb{R}^{m \times (pn+p)} \qquad W \in \mathbb{R}^{m \times m}$$

We accepted both **sets** of answers as correct.

(b) [2 points] Let $\theta \in \mathbb{R}^N$. Define $\Gamma \in \mathbb{R}^{N_0 \times N}$ to be any matrix. Suppose we add a term $P(\theta)$ to our cost function:

$$P(\theta) = \frac{1}{2}\sum_{i=1}^{N_0}(\Gamma\theta)_i^2.$$

Show that $P(\theta)$ can be written as

$$P(\theta) = \frac{1}{2}\mathrm{tr}((\Gamma\theta)^T(\Gamma\theta)) = \frac{1}{2}\left\|\Gamma\theta\right\|_2^2.$$

Answer: The definition of $\|u\|_2^2 = \sum_{i=1}^n u_i^2$. So $\sum_{i=1}^{N_0}(\Gamma\theta)_i^2 = \|\Gamma\theta\|_2^2$, which is $(\Gamma\theta)^T(\Gamma\theta) \in \mathbb{R}_+$. And $\mathrm{tr}(a) = a$ for any scalar.

(c) [4 points] Our final cost function is:

$$J(\theta) = \frac{1}{2}\mathrm{tr}\Big[(X\theta - y)^T W(X\theta - y)\Big] + \frac{1}{2}\mathrm{tr}((\Gamma\theta)^T(\Gamma\theta)) \tag{1}$$

Derive a closed form expression for the minimizer θ^* that minimizes $J(\theta)$ as shown in Equation (1).

Answer: We compute the gradient of the first and second term separately. We start with the first term, $J_1(\theta) = \frac{1}{2}\mathrm{tr}\left((X\theta - y)^T W(X\theta - y)\right)$.

$$\begin{aligned}
\nabla_\theta J_1(\theta) &= \nabla_\theta \frac{1}{2}\mathrm{tr}\left((X\theta - y)^T W(X\theta - y)\right) \\
&= \frac{1}{2}\nabla_\theta \mathrm{tr}\Big(\theta^T X^T W X\theta - \theta^T X^T W y - y^T W X\theta - y^T W y\Big) \\
&= \frac{1}{2}\nabla_\theta \left[\mathrm{tr}(\theta^T X^T W X\theta) - \mathrm{tr}(\theta^T X^T W y) - \mathrm{tr}(y^T W X\theta) - \mathrm{tr}(y^T W y)\right] \\
&= \frac{1}{2}\nabla_\theta \left[\mathrm{tr}(\theta^T X^T W X\theta) - 2\mathrm{tr}(y^T W X\theta) - \mathrm{tr}(y^T W y)\right] \\
&= \frac{1}{2}(X^T W X\theta - 2X^T W y + X^T W X\theta) \\
&= X^T W X\theta - X^T W y
\end{aligned}$$

Now we find the gradient of the second term, $J_2(\theta) = \frac{1}{2}\mathrm{tr}((\Gamma\theta)^T(\Gamma\theta))$:

$$\begin{aligned}
\nabla_\theta J_2(\theta) &= \frac{1}{2}\nabla_\theta \mathrm{tr}((\Gamma\theta)^T(\Gamma\theta)) \\
&= \frac{1}{2}\nabla_\theta \mathrm{tr}(\theta^T\Gamma^T\Gamma\theta) \\
&= \frac{1}{2}(2\Gamma^T\Gamma\theta) \\
&= \Gamma^T\Gamma\theta
\end{aligned}$$

Combining the gradient of both terms gives us the final gradient:

$$\begin{aligned}\nabla_\theta J(\theta) &= \nabla_\theta J_1(\theta) + \nabla_\theta J_2(\theta) \\ &= X^T W X\theta - X^T W y + \Gamma^T \Gamma\theta\end{aligned}$$

We can then set $\nabla_\theta J(\theta)$ equal to zero and find the optimal θ^*:

$$\begin{aligned}0 &= X^T W X\theta - X^T W y + \Gamma^T \Gamma\theta \\ X^T W y &= X^T W X\theta + \Gamma^T \Gamma\theta \\ X^T W y &= (X^T W X + \Gamma^T \Gamma)\theta \\ \theta^* &= (X^T W X + \Gamma^T \Gamma)^{-1} X^T W y\end{aligned}$$

Aside: Let I be the $n \times n$ identity matrix. If $\Gamma = \alpha I$ for some $\alpha > 0$, the above technique is known as *ridge regression* or ℓ_2 regularization.

(d) [2 points] If we want to maximize the *training* accuracy, what is the optimal value of Γ (if any)? In 1-2 sentences, justify your answer.

Answer: Choose $\Gamma = 0$, as this means we choose θ exclusively to minimize the training error.

(e) [2 points] If we want to maximize the *test* accuracy, what is the optimal value of Γ (if any)? In 1-2 sentences, justify your answer.

Answer: It depends. There is no particular value that is guaranteed to minimize test accuracy.

(f) [2 points] So far, we used a regression model containing polynomial representations of the input. Our polynomial model contains ${\theta^{(1)}}^T x$ as a term which is the same as our "standard" linear model of $y = \theta^T x$. However, our polynomial model can express higher-order relationships while our standard model cannot. In 2-4 sentences, explain when and why we should *not* use the polynomial model.

Answer: Consider a linearly separable dataset (i.e. the original, non-polynomial $x^{(i)}$'s are good enough to predict $y^{(i)}$). Including polynomial terms is unnecessary, since we know the dataset can be modeled without them. If p is large, this will increase the VC dimension which can lead to overfitting. In this case, we should not use the polynomial model.

4. **[14 points] Online (not stochastic) gradient descent**

   In this question, we explore a variant of stochastic gradient descent known as *online* gradient descent. A cost function $c : \mathbb{R}^n \to \mathbb{R}$ is *convex* if

   $$c(\lambda\theta + (1-\lambda)\overline{\theta}) \leq \lambda c(\theta) + (1-\lambda)c(\overline{\theta})$$

   for all $\theta, \overline{\theta} \in \mathbb{R}^n$. A differentiable convex function c satisfies

   $$c(\overline{\theta}) \geq c(\theta) + \nabla c(\theta)^T(\overline{\theta} - \theta) \quad \text{for all } \overline{\theta} \in \mathbb{R}^n.$$

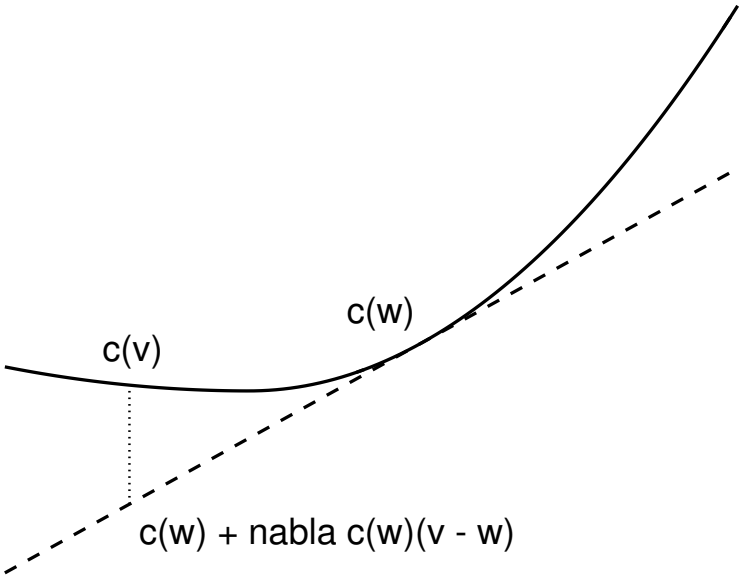


Figure 3: A convex function and its linear approximation at the point θ.

In online convex optimization, the learner receives (sequentially) a sequence of convex functions $c_1, c_2, c_3, \ldots$, and at iteration t makes the online gradient update

$$\theta^{(t+1)} = \theta^{(t)} - \alpha g^{(t)} \quad \text{where} \quad g^{(t)} = \nabla c_t(\theta^{(t)}). \tag{2}$$

Here $\alpha > 0$ is a scalar stepsize, and we assume that all cost functions c_t are differentiable. The goal is to not suffer too much cumulative loss $\sum_{t=1}^{T} c_t(\theta^{(t)})$.

(a) [4 points] Prove that with the update (2), for any $\theta \in \mathbb{R}^n$,

$$\frac{1}{2}\left\|\theta^{(t+1)} - \theta\right\|_2^2 \leq \frac{1}{2}\left\|\theta^{(t)} - \theta\right\|_2^2 - \alpha(c_t(\theta^{(t)}) - c_t(\theta)) + \frac{\alpha^2}{2}\left\|g^{(t)}\right\|_2^2.$$

Answer: We expand $\theta^{(t+1)}$ in terms of $\theta^{(t)}$, getting

$$\begin{aligned}\frac{1}{2}\left\|\theta^{(t+1)} - \theta\right\|_2^2 &= \frac{1}{2}\left\|\theta^{(t)} - \alpha g^{(t)} - \theta\right\|_2^2 \\ &= \frac{1}{2}\left\|\theta^{(t)} - \theta\right\|_2^2 - \alpha {g^{(t)}}^T(\theta^{(t)} - \theta) + \frac{\alpha^2}{2}\left\|g^{(t)}\right\|_2^2.\end{aligned}$$

Then use convexity to see that

$$-{g^{(t)}}^T(\theta^{(t)} - \theta) = -\nabla c_t(\theta^{(t)})^T(\theta^{(t)} - \theta) = \nabla c_t(\theta^{(t)})^T(\theta - \theta^{(t)}) \leq c_t(\theta) - c_t(\theta^{(t)}).$$

We substitute to obtain

$$\frac{1}{2}\left\|\theta^{(t+1)} - \theta\right\|_2^2 \leq \frac{1}{2}\left\|\theta^{(t)} - \theta\right\|_2^2 + \alpha(c_t(\theta) - c_t(\theta^{(t)})) + \frac{\alpha^2}{2}\left\|g^{(t)}\right\|_2^2.$$

(b) [4 points] After T iterations of online gradient descent (2), the *regret* of the learner with respect to a fixed $\theta \in \mathbb{R}^n$ is

$$\mathsf{Reg}_T(\theta) := \sum_{t=1}^{T}\left[c_t(\theta^{(t)}) - c_t(\theta)\right].$$

Using the result of part (a), show that

$$\mathsf{Reg}_T(\theta) = \sum_{t=1}^{T}\left[c_t(\theta^{(t)}) - c_t(\theta)\right] \leq \frac{1}{2\alpha}\left\|\theta^{(1)} - \theta\right\|_2^2 + \frac{\alpha}{2}\sum_{t=1}^{T}\left\|g^{(t)}\right\|_2^2.$$

Answer: Rearrange the result of part (a) to find that

$$c_t(\theta^{(t)}) - c_t(\theta) \leq \frac{1}{2\alpha}\left[\left\|\theta^{(t)} - \theta\right\|_2^2 - \left\|\theta^{(t+1)} - \theta\right\|_2^2 + \alpha^2\left\|g^{(t)}\right\|_2^2\right].$$

Sum this expression from $t = 1$ to $t = T$, which gives

$$\begin{aligned}\sum_{t=1}^{T}\left[c_t(\theta^{(t)}) - c_t(\theta)\right] &\leq \frac{1}{2\alpha}\left[\left\|\theta^{(1)} - \theta\right\|_2^2 - \left\|\theta^{(T+1)} - \theta\right\|_2^2\right] + \frac{\alpha}{2}\sum_{t=1}^{T}\left\|g^{(t)}\right\|_2^2 \\ &\leq \frac{\left\|\theta^{(1)} - \theta\right\|_2^2}{2\alpha} + \frac{\alpha}{2}\sum_{t=1}^{T}\left\|g^{(t)}\right\|_2^2.\end{aligned}$$

(c) [4 points] Suppose we guarantee that the functions c_t have *bounded* gradients, that is, $\left\|g^{(t)}\right\|_2 \leq G$ for all t. Give a stepsize α, which may depend on $\|\theta\|_2$ and G, such that if $\theta^{(1)} = 0$, we can guarantee

$$\mathsf{Reg}_T(\theta) \leq G\|\theta\|_2\sqrt{T}.$$

That is, the *average* regret $\frac{1}{T}\mathsf{Reg}_T(\theta) = O(1/\sqrt{T})$ for any vector θ.

Answer: Set $\alpha = \frac{\|w\|_2}{G\sqrt{T}}$ and get

$$\mathsf{Reg}_T(w) \le \frac{\|w\|_2^2}{2\alpha} + \frac{\alpha T G^2}{2} = \frac{\|w\|_2 G}{2\sqrt{T}} + \frac{G\|w\|_2}{2\sqrt{T}} = G\|w\|_2\sqrt{T}.$$

(d) [2 points] Show that if $y \in \{-1, 1\}$ and x satisfies $\|x\|_2 \le G$, then the gradient of the logistic loss (the logistic loss is $\mathsf{L}(\theta^T x, y) = \log(1 + \exp(-y\theta^T x))$) has ℓ_2-norm bounded by G.

Answer: Taking derivatives of logistic regression, we have

$$\nabla_\theta \log(1 + \exp(-y\theta^T x)) = -\frac{1}{1 + e^{y\theta^T x}}(yx).$$

Taking the ℓ_2-norm of this, we have

$$\left\|\nabla_\theta \log(1 + \exp(-y\theta^T x))\right\|_2 = \underbrace{\frac{1}{1 + e^{y\theta^T x}}}_{\le 1} \|x\|_2 \le \|x\|_2 \le G.$$

5. **[17 points] Kernels via randomization**

You have seen how using kernels can allow efficient predictions by using the representer theorem, and the kernel trick allows us to automatically incorporate nonlinearities in supervised learning problems via the kernel function K. A difficulty with kernels is their time complexity: if we form the kernel (Gram) matrix G,[1] defined by

$$G_{ji} = G_{ij} = K(x^{(i)}, x^{(j)}), \quad G \in \mathbb{R}^{m \times m},$$

then storing G requires space $O(m^2)$, inverting it requires time $O(m^3)$, and making new predictions on an unseen point x requires time $m \cdot T$, where T is the amount of time to compute $K(x, x^{(i)})$. One way around this is via randomization.

Suppose that the raw input attributes $x \in \mathcal{X}$, and let $\mathcal{W}$ be some other space (you may assume that $\mathcal{W} = \mathbb{R}$). Let $\phi : \mathcal{X} \times \mathcal{W} \to \mathbb{R}$ be an arbitrary function, and let P be a probability distribution on the space $\mathcal{W}$. Define the function

$$K_P(x, z) := \mathbb{E}[\phi(x, W)\phi(z, W)] \quad \text{for } x, z \in \mathcal{X}, \tag{3}$$

where the subscript P denotes that W is sampled according to P (i.e. the expectation is taken over $W \sim P$).

(a) [4 points] Is the function K_P a valid (Mercer) kernel? If so, prove this. If not, give a counterexample.

Answer: The function K_P is indeed a Mercer kernel. Indeed, define the Gram matrix G by $G_{ij} = K_P(x^{(i)}, x^{(j)})$. Then it is clear that $G_{ij} = G_{ji}$, and for any vector $v \in \mathbb{R}^m$, we have

$$\begin{aligned} v^T G v &= \sum_{i,j} v_i v_j G_{ij} = \sum_{i,j} v_i \mathbb{E}[\phi(x^{(i)}, W)\phi(x^{(j)}, W)] v_j \\ &= \mathbb{E}\left[\sum_{i,j} v_i \phi(x^{(i)}, W)\phi(x^{(j)}, W) v_j\right] \\ &= \mathbb{E}\left[\left(\sum_{i=1}^{m} \phi(x^{(i)}, W) v_i\right)^2\right] \geq 0. \end{aligned}$$

In particular, the matrix G is positive semidefinite, so that K_P is a valid kernel.

[1] We use G in this problem so as not to confuse it with K, the kernel function

(b) [4 points] A natural idea is to approximate K_P by random sampling. We take N i.i.d. samples $W_l \overset{\text{iid}}{\sim} P$, calling them $W_1, W_2, \ldots, W_N$, and we define

$$\widehat{K}(x,z) := \frac{1}{N}\sum_{l=1}^{N} \phi(x,W_l)\phi(z,W_l).$$

Suppose we know that $\phi(x,w) \in [-1,1]$ for all $x \in \mathcal{X}$ and all $w \in \mathcal{W}$. For a fixed pair $x, z \in \mathcal{X}$, give an upper bound on the probability that $\widehat{K}$ is far from K_P, that is, give a bound decreasing to 0 *exponentially* in N on

$$\mathbb{P}\left(\left|\widehat{K}(x,z) - K_P(x,z)\right| \geq \epsilon\right)$$

that is valid for all $\epsilon \geq 0$.

Answer: We use Hoeffding's inequality. In particular, if we define the random variable

$$Z_l = \phi(x,W_l)\phi(z,W_l)$$

then $Z_l \in [-1,1]$, and $\mathbb{E}[Z_l] = \mathbb{E}[\phi(x,W)\phi(z,W)] = K_P(x,z)$. Hoeffding's inequality then implies that

$$\begin{aligned}\mathbb{P}(\widehat{K}(x,z) - K_P(x,z) \geq \epsilon) &= \mathbb{P}\left(\frac{1}{N}\sum_{l=1}^{N}(Z_l - \mathbb{E}[Z_l]) \geq \epsilon\right) \\ &\leq \exp\left(-\frac{2N\epsilon^2}{(1+1)^2}\right) = \exp\left(-\frac{N\epsilon^2}{2}\right)\end{aligned}$$

and similarly for the event that $\widehat{K}(x,z) - K_P(x,z) \leq -\epsilon$. Thus

$$\mathbb{P}\left(\left|\widehat{K}(x,z) - K_P(x,z)\right| \geq \epsilon\right) \leq 2\exp\left(-\frac{N\epsilon^2}{2}\right).$$

(c) [4 points] Continue to assume that $\phi(x, w) \in [-1, 1]$ for all x, w. Suppose we have a training set $\{x^{(i)}\}_{i=1}^m$ of size m. Give a sample size N^* such that if we take $N \geq N^*$ samples of W we are guaranteed that with probability at least $1 - \delta$, we have

$$\left| \widehat{K}(x^{(i)}, x^{(j)}) - K_P(x^{(i)}, x^{(j)}) \right| \leq \epsilon$$

for all pairs $i, j \in \{1, \ldots, m\}$. Written differently, if $\widehat{G}_{ij} = \widehat{K}(x^{(i)}, x^{(j)})$ and $G_{ij} = K_P(x^{(i)}, x^{(j)})$, guarantee that $\max_{i,j} |\widehat{G}_{ij} - G_{ij}| \leq \epsilon$.

Answer:

As the kernel functions $\widehat{K}$ and K_P are symmetric, we need concern ourselves only with indices $i \leq j$, of which there are $\frac{m(m+1)}{2}$. By the union bound, we have

$$\begin{aligned}
\mathbb{P}\left(\max_{i,j} \left| \widehat{G}_{ij} - G_{ij} \right| \geq \epsilon \right) &\leq \sum_{i \leq j}^{m} \mathbb{P}\left(\left| \widehat{K}(x^{(i)}, x^{(j)}) - K_P(x^{(i)}, x^{(j)}) \right| \geq \epsilon \right) \\
&\leq m(m+1) \exp\left(-\frac{N\epsilon^2}{2} \right)
\end{aligned}$$

by part (b). Setting this value equal to δ, we solve to obtain

$$\delta = m(m+1) \exp\left(-\frac{N\epsilon^2}{2} \right) \quad \text{iff} \quad \frac{N\epsilon^2}{2} = \log \frac{m(m+1)}{\delta},$$

so that it is sufficient that we have

$$N \geq \frac{2 \log \frac{m^2+m}{\delta}}{\epsilon^2}$$

samples of W to guarantee good approximation for all pairs $x^{(i)}, x^{(j)}$.

(d) [5 points] Assume that you have N i.i.d. samples $W_1, \ldots, W_N \overset{\text{iid}}{\sim} P$ and a training set $\{(x^{(i)}, y^{(i)})\}_{i=1}^m$ for a binary classification problem, with $y^{(i)} \in \{-1, 1\}$, and loss $\mathsf{L} : \mathbb{R} \times \{-1, 1\} \to \mathbb{R}$. In the usual kernelized supervised learning setting, we would make predictions on a new datapoint x using $\sum_{i=1}^m K_P(x, x^{(i)})\alpha_i$, and if

$$G = \begin{bmatrix} G^{(1)} & \cdots & G^{(m)} \end{bmatrix} \in \mathbb{R}^{m \times m}, \quad G^{(i)} \in \mathbb{R}^m$$

is the Gram matrix, we would choose α by minimizing

$$J_\lambda(\alpha) = \frac{1}{m} \sum_{i=1}^m \mathsf{L}({G^{(i)}}^T \alpha, y^{(i)}) + \frac{\lambda}{2} \alpha^T G \alpha. \tag{4}$$

Using your N samples $W_1, \ldots, W_N$, how can we *reverse* the kernel trick? That is, (i) write down a supervised learning problem with optimization variable $\theta \in \mathbb{R}^N$ that approximates problem (4), (ii) describe how, when given a new datapoint x, you can make a prediction on that datapoint, and (iii) give a bound on the runtime of making a prediction on a new datapoint x.

Answer: (i) We define the vector

$$\widehat{\phi}(x) := \begin{bmatrix} \phi(x, W_1) \\ \vdots \\ \phi(x, W_N) \end{bmatrix}.$$

Then we let $\theta \in \mathbb{R}^n$, and write the regularized risk

$$J_\lambda(\theta) = \frac{1}{m} \sum_{i=1}^m \mathsf{L}(\widehat{\phi}(x^{(i)})^T \theta, y^{(i)}) + \frac{\lambda}{2} \|\theta\|_2^2.$$

The representer theorem tells us that this is equivalent to minimizing

$$\widehat{J}_\lambda(\alpha) = \frac{1}{m} \sum_{i=1}^m \mathsf{L}((\widehat{G}^{(i)})^T \alpha, y^{(i)}) + \frac{\lambda}{2} \alpha^T \widehat{G} \alpha,$$

which approximates $J_\lambda(\alpha)$ because we have $\widehat{G} \approx G$ by sampling. (ii) We make predictions on a new point x via

$$h_\theta(x) = \widehat{\phi}(x)^T \theta = \sum_{l=1}^N \phi(x, W_l)\theta_l.$$

(iii) The runtime of the predictions is $O(NT_\phi)$, where T_ϕ is the time to compute $\phi(x, w)$.

6. **[30 points] Linear Regression and Boosting**

In this problem, we consider boosting for regression, where we combine weak predictors $\phi : \mathcal{X} \to \{-1, 1\}$ to predict real-valued targets $y^{(i)} \in \mathbb{R}$. To handle regression, we use the least-squares cost function,

$$J(\theta) = \sum_{i=1}^{m} \left(\Phi(x^{(i)})^T\theta - y^{(i)}\right)^2$$

where θ is our parameter vector and $\Phi(x)$ is our feature vector. We assume the training examples $x^{(i)} \in \mathbb{R}$, and that all of the values $x^{(i)}$ are unique.

You will derive a new boosting update, derive an analogue of the edge used in binary classification, and show how to construct decision stumps.

Here is some notation for the iterative boosting procedure, where $t \in \{1, 2, 3, \ldots\}$ indicates the iteration of boosting:

$\theta^{(t)} = [\theta_1\ \theta_2\ \cdots\ \theta_t]^T \in \mathbb{R}^t$	[parameter vector at time t]
$\phi_i^{(t)} = \phi^{(t)}(x^{(i)})$	[t^{th} weak learner applied to example i]
$\Phi_i^{(t)} = [\phi_i^{(1)}\ \cdots\ \phi_i^{(t)}]^T \in \mathbb{R}^t$	[vector of weak learners for example i]
$\phi^{(t)} = [\phi_1^{(t)}\ \cdots\ \phi_m^{(t)}]^T \in \mathbb{R}^m$	[weak learner t applied to each example]
$h_i^{(t)} = (\Phi_i^{(t)})^T\theta^{(t)}$	[current prediction for example i]

We can compactly write the value of of the cost function at time step t as

$$J^{(t)} = J(\theta^{(t)}) = \sum_{i=1}^{m} \left(h_i^{(t)} - y^{(i)}\right)^2 = \left\|h^{(t)} - y\right\|_2^2,$$

where $h^{(t)} \in \mathbb{R}^m$ is the vector of predictions at iteration t of the boosting procedure for *all* training examples and $y = [y^{(1)}\ \cdots\ y^{(m)}]^T \in \mathbb{R}^m$.

(a) [3 points] Express $J^{(t)}$ in terms of $h^{(t-1)}$, $\phi^{(t)}$, and θ_t instead of $h^{(t)}$. [*Hint*: express $h_i^{(t)}$ in terms of $h_i^{(t-1)}$. Then express $h^{(t)}$ in terms of $h^{(t-1)}$ using $\phi^{(t)}$.]

Answer:

$$h_i^{(t)} = h_i^{(t-1)} + \phi_i^{(t)}\theta_t, \quad \text{so} \quad h^{(t)} = h^{(t-1)} + \phi^{(t)}\theta_t$$

where $\phi^{(t)} \in \mathbb{R}^m$. Then

$$J^{(t)} = \left\|h^{(t)} - y\right\|_2^2 = \left\|h^{(t-1)} + \phi^{(t)}\theta_t - y\right\|_2^2.$$

(b) [3 points] Let h, y, and ϕ be vectors in $\mathbb{R}^m$. What value of α minimizes $\|h - y + \phi\alpha\|_2^2$, where $\alpha \in \mathbb{R}$ is a 1-dimensional scalar?

Answer: We expand the square in $\|h - y + \phi\alpha\|_2^2$, which gives

$$\begin{aligned}
\|h - y + \phi\alpha\|_2^2 &= (h - y + \phi\alpha)^T(h - y + \phi\alpha) \\
&= h^Th - h^Ty + h^T\phi\alpha - y^Th + y^Ty - y^T\phi\alpha + \alpha\phi^Th - \alpha\phi^Ty + \alpha^2\phi^T\phi \\
&= h^Th - 2h^Ty + 2h^T\phi\alpha + y^Ty - 2y^T\phi\alpha + \alpha^2\phi^T\phi.
\end{aligned}$$

Taking derivatives we obtain

$$\begin{aligned}\frac{\partial}{\partial\alpha}\left\|h-y+\phi\alpha\right\|_2^2 &= 2h^T\phi - 2y^T\phi + 2\alpha\phi^T\phi \\ &= \phi^T(2h-2y) + 2\alpha\left\|\phi\right\|_2^2.\end{aligned}$$

By setting to zero and isolating α, we find that

$$\alpha = \phi^T(y-h)/\left\|\phi\right\|_2^2$$

(c) [3 points] We have performed boosting for $t-1$ iterations, and wish to add the tth weak predictor, with predictions $\phi^{(t)} = [\phi^{(t)}(x^{(1)}) \ \cdots \ \phi^{(t)}(x^{(m)})]^T \in \mathbb{R}^m$. What is the optimal value for θ_t in terms of $h^{(t-1)}$, y, and $\phi^{(t)}$? [*Hint*: see part (6b).]

Answer: Using the result of part (b), we have

$$\theta_t = (\phi^{(t)})^T(y-h^{(t-1)})/\left\|\phi^{(t)}\right\|_2^2.$$

(d) [4 points] Write an expression for the minimal value $\min_\alpha \|h - y + \phi\alpha\|_2^2$ in terms of h, y, ϕ. [*Hint*: if $I \in \mathbb{R}^{m\times m}$ is the identity matrix and $u \in \mathbb{R}^m$ satisfies $\|u\|_2 = 1$, then what does $(I - uu^T)^2 = (I - uu^T)^T(I - uu^T)$ equal?]

Answer: First, we show that $(I-uu^T)$ is a symmetric idempotent matrix, meaning that applying it twice is the same as applying it once. We have

$$\begin{aligned}(I - uu^T)^T(I - uu^T) &= (I - uu^T)(I - uu^T)\\ &= I - 2uu^T + uu^Tuu^T = I - 2uu^T + u\|u\|_2^2 u^T\\ &= I - 2uu^T + uu^T = I - uu^T.\end{aligned}$$

Now, writing the value for the optimal α from the previous part, we have

$$h - y + \phi\alpha = h - y + \frac{1}{\|\phi\|_2^2}\phi\phi^T(y - h) = \left(I_{m\times m} - \frac{\phi}{\|\phi\|_2}\frac{\phi^T}{\|\phi\|_2}\right)(h - y).$$

Let $u = \phi/\|\phi\|_2$. Then

$$\begin{aligned}\min_\alpha \|h - y + \phi\alpha\|_2^2 &= \left\|(I - uu^T)(h - y)\right\|_2^2\\ &= (h - y)^T(I - uu^T)^T(I - uu^T)(h - y)\\ &= (h - y)^T(I - uu^T)(h - y)\\ &= \|h - y\|_2^2 - (h - y)^Tuu^T(h - y)\\ &= \|h - y\|_2^2 - \left(\frac{1}{\|\phi\|_2}\phi^T(h - y)\right)^2.\end{aligned}$$

(e) [5 points] Let $J^{(t)}$ be the minimal value of the cost after adding the tth feature function $\phi^{(t)}$ and parameter θ_t, that is,

$$J^{(t)} = \min_{\theta_t} \sum_{i=1}^m \left(\Phi_i^{(t-1)^T}\theta^{(t-1)} + \theta_t\phi^{(t)}(x^{(i)}) - y^{(i)}\right)^2.$$

Find $\delta^{(t)}$ such that $J^{(t)} = J^{(t-1)} - \delta^{(t)}$, where $\delta^{(t)}$ is a function of $h^{(t-1)}$, y, and $\phi^{(t)}$. [*Hint*: see part (6d).]

Answer:

$$\begin{aligned}J^{(t)} &= \min_\alpha \left\|h^{(t-1)} - y + \phi^{(t)}\alpha\right\|_2^2\\ &= \left\|h^{(t-1)} - y\right\|_2^2 - \frac{1}{\|\phi^{(t)}\|_2^2}\left((\phi^{(t)})^T(h^{(t-1)} - y)\right)^2\\ &= J^{(t-1)} - \frac{1}{\|\phi^{(t)}\|_2^2}\left((\phi^{(t)})^T(h^{(t-1)} - y)\right)^2.\end{aligned}$$

We can simplify this slightly because we are using sign functions, which satisfy $\|\phi\|_2^2 = m$:

$$J^{(t)} = J^{(t-1)} - \underbrace{\frac{1}{m}\left((\phi^{(t)})^T(h^{(t-1)} - y)\right)^2}_{=\delta^{(t)}}.$$

We have a decrease in the cost function: $J^{(t)} \le J^{(t-1)}$.

(f) [6 points] Let $h \in \mathbb{R}^m$ be an arbitrary vector and let $y \in \mathbb{R}^m$. Suppose that we use thresholded decision stumps on $x \in \mathbb{R}$, so that $\phi(x) = \text{sign}(x - s)$ for some $s \in \mathbb{R}$. (Here we let $\text{sign}(\beta) = 1$ for $\beta \geq 0$ and $\text{sign}(\beta) = -1$ for $\beta < 0$.) Consider the following expression:

$$F(s) := \sum_{i=1}^{m} \text{sign}(x_i - s)(h_i - y^{(i)}),$$

where you may assume that $x_1 > x_2 > \cdots > x_m$. Define

$$f(m_0) = \sum_{i=1}^{m_0} (h_i - y^{(i)}) - \sum_{i=m_0+1}^{m} (h_i - y^{(i)}).$$

Show that for each s, there is some $m_0(s)$ such that $F(s) = f(m_0(s))$. Additionally, show there is some s such that

$$|F(s)| \geq \max_i |y^{(i)} - h_i| = \|y - h\|_\infty .$$

[*Hint:* The boosting techniques of PS2 may be useful.]

Answer: We take $m_0(s)$ to be the smallest index i such that $x_i \geq s$. Then $\text{sign}(x_i - s) = 1$ for $i \leq m_0$ and $\text{sign}(x_i - s) = -1$ for $i > m_0$, which shows that $F(s) = f(m_0(s))$. For the second part, we note that for any m_0, we have

$$f(m_0) - f(m_0 - 1) = 2(h_{m_0} - y_{m_0}),$$

and thus there must be *some* index m^* such that $|f(m^*)| \geq \max_i |y^{(i)} - h_i|$. Choosing s so that $x_i < s$ for $i \geq m^*$ and $x_i \geq s$ for $i < m^*$ gives the result.

(g) [6 points] A sufficiently good weak-learning procedure, at iteration t, choose a threshold s, which gives a thresholded stump $\phi^{(t)}(x) = \text{sign}(x - s)$, such that

$$\left| \sum_{i=1}^{m} \phi^{(t)}(x^{(i)})(h_i^{(t-1)} - y^{(i)}) \right| = \left| \sum_{i=1}^{m} \text{sign}(x^{(i)} - s)(h_i^{(t-1)} - y^{(i)}) \right| \geq \left\| h^{(t-1)} - y \right\|_\infty = \max_i |h_i^{(t-1)} - y^{(i)}|. \quad (5)$$

i. [1 points] Use part (6f) to argue that there is a weak-learning procedure for which the guarantee (5) holds. (Assume the $x^{(i)} \in \mathbb{R}$ are all unique.)

ii. [5 points] Assuming we have the guarantee (5), give a value $\gamma > 0$, which may depend on m, such that

$$J^{(t)} \leq (1 - \gamma) J^{(t-1)}.$$

By part (6e), this is equivalent to showing that $\delta^{(t)} \geq \gamma J^{(t-1)}$. State explicitly what your γ is. [*Hint:* for a vector $v \in \mathbb{R}^m$, we have $\sqrt{m} \|v\|_\infty \geq \|v\|_2$, where $\|v\|_\infty = \max_i |v_i|$. Also $\|\phi^{(t)}\|_2^2 = m$ because $\phi^{(t)} \in \{-1, 1\}^m$.]

Answer: The guarantee (5) is immediate by part (6f), because we simply choose the best s for any given attribute index j.

Now we use the definition of $\delta^{(t)}$ in part (6e). We have

$$\delta^{(t)} = \frac{1}{m} \left((\phi^{(t)})^T (h^{(t-1)} - y) \right)^2 = \frac{1}{m} \left(\sum_{i=1}^{m} \phi^{(t)}(x^{(i)})(h_i^{(t-1)} - y^{(i)}) \right)^2.$$

Recalling that

$$\left| \sum_{i=1}^{m} \phi^{(t)}(x^{(i)})(h_i^{(t-1)} - y^{(i)}) \right| \geq \left\| y - h^{(t-1)} \right\|_\infty,$$

we obtain

$$\begin{aligned} \delta^{(t)} &= \left(\frac{1}{\sqrt{m}} \sum_{i=1}^{m} \phi_i^{(t)}(h_i^{(t)} - y_i) \right)^2 \\ &\geq \left(\frac{1}{\sqrt{m}} \left\| y - h^{(t-1)} \right\|_\infty \right)^2 \\ &= \frac{1}{m} \left\| y - h^{(t-1)} \right\|_\infty^2 \\ &\geq \frac{1}{m^2} \left\| y - h^{(t-1)} \right\|_2^2. \end{aligned}$$

So $\gamma = \frac{1}{m^2}$, where the third equality follows by our setting of $\phi_i^{(t)}$.