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

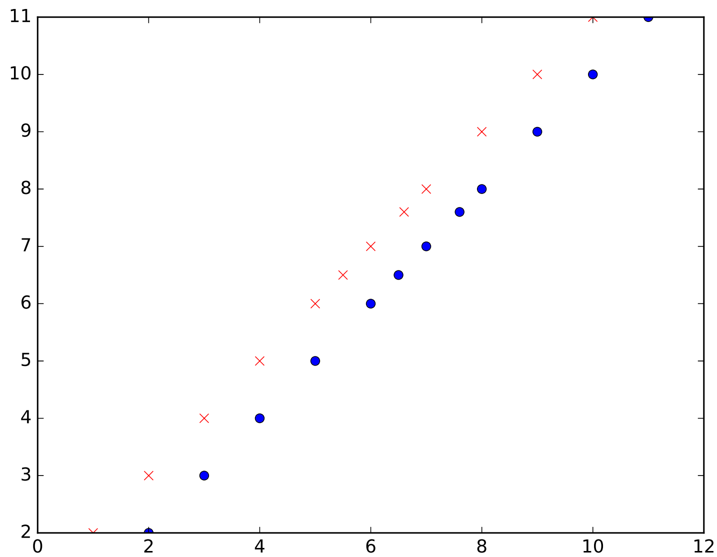
The Stanford University Honor Code:

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: ____________________________________

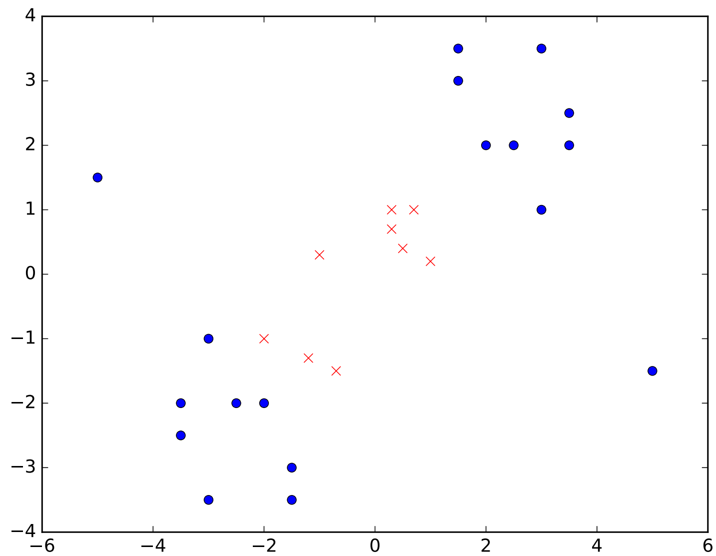
1. **[21 points] Short Answer**

   (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.



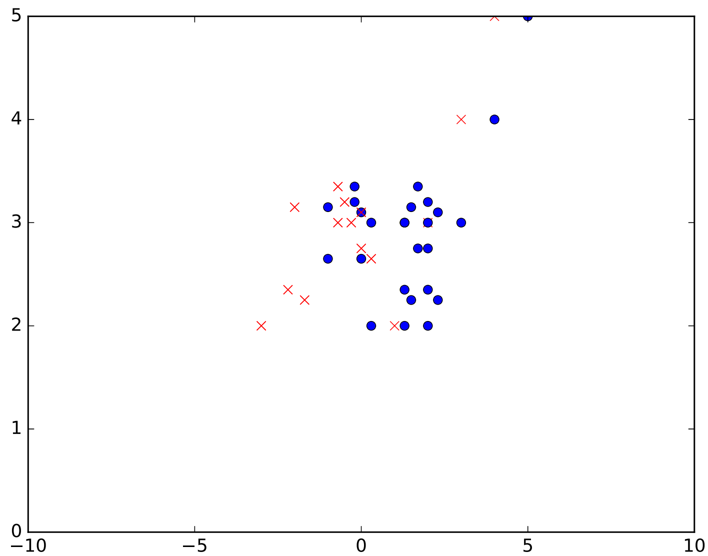
A supervised learning method that would likely work:

A supervised learning method that would likely not work:



A supervised learning method that would likely work:

A supervised learning method that would likely not work:



A supervised learning method that would likely work:

A supervised learning method that would likely not work:

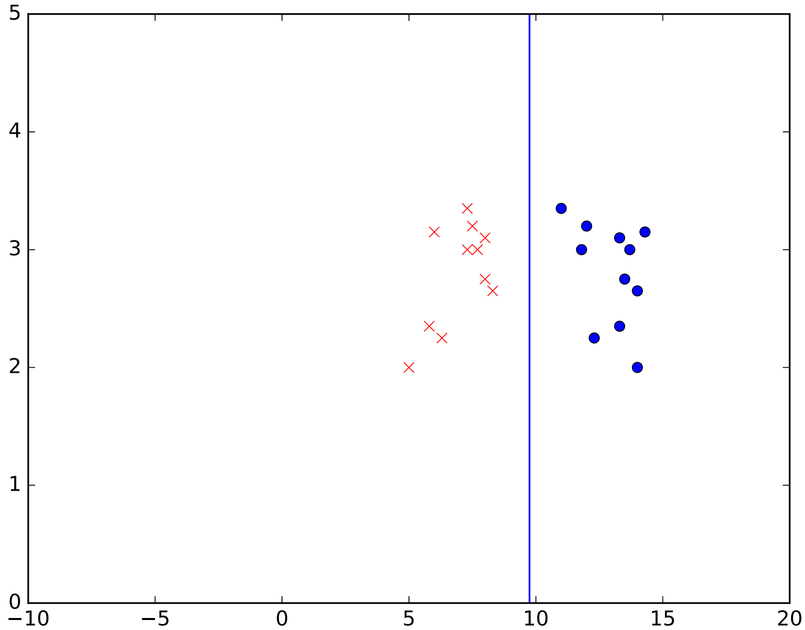


Figure 1: An easy to separate dataset.

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

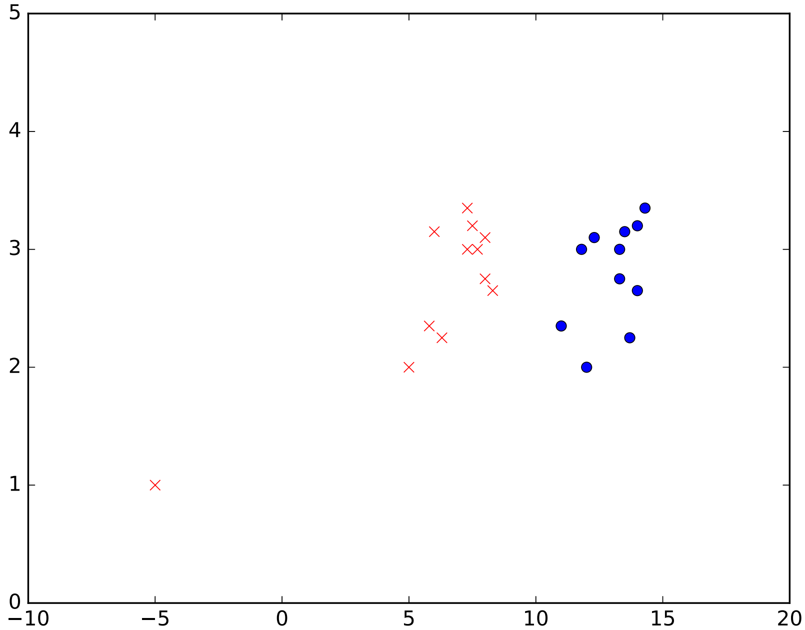


Figure 2: A new point.

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.

(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?

(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

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

no matter the training set size m.

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

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

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.

(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."]

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.

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

   We have a training set:

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

   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.

(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

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

(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).

(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.

(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.

(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.

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

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

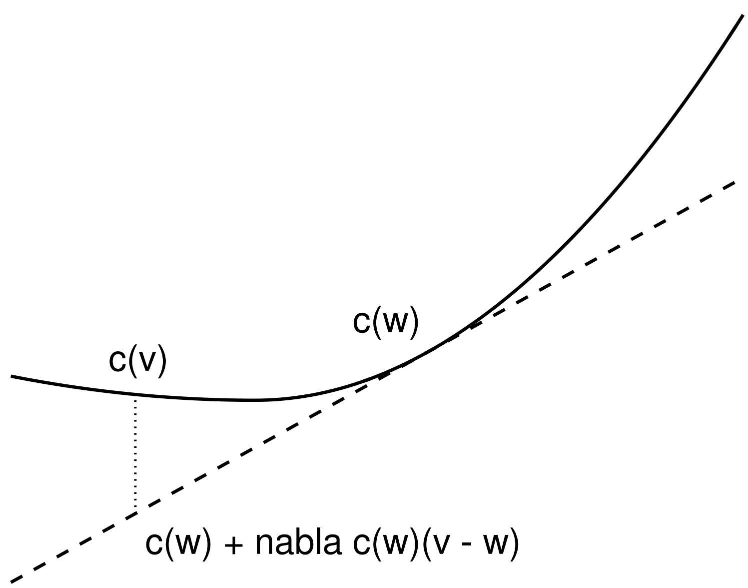


Figure 3: A convex function and its linear approximation at the point θ.

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 $\theta \in \mathbb{R}^n$,

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

(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

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

(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

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

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

(d) [2 points] Show that if $y \in \{-1, 1\}$ and x satisfies $\|x\|_2 \leq G$, 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.

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.

[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 $W_l \stackrel{\text{iid}}{\sim} P$, 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 $\epsilon \geq 0$.

(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 $\max_{i,j} |\widehat{G}_{ij} - G_{ij}| \leq \epsilon$.

(d) [5 points] Assume that you have N i.i.d. samples $W_1, \ldots, W_N \stackrel{\text{iid}}{\sim} P$ 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.

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:

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

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 $\phi^{(t)}$.]

(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?

(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).]

(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?]

(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).]

(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 $\phi(x) = \mathrm{sign}(x - s)$ for some $s \in \mathbb{R}$. (Here we let $\mathrm{sign}(\beta) = 1$ for $\beta \geq 0$ and $\mathrm{sign}(\beta) = -1$ for $\beta < 0$.) Consider the following expression:

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

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.]

(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

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

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 $\sqrt{m}\left\|v\right\|_\infty \geq \left\|v\right\|_2$, where $\left\|v\right\|_\infty = \max_i |v_i|$. Also $\|\phi^{(t)}\|_2^2 = m$ because $\phi^{(t)} \in \{-1, 1\}^m$.]