Theorem[section]

DS-GA-1003: Machine Learning (Spring 2020)

Midterm Exam (March 10 5:20-7:00PM)

- You have **90 minutes** to complete the exam. No textbooks, notes, computers or calculators. However you are allowed a double-sided reference sheet.
- The exam consists of 9 single-sided pages. Mark your answers on the exam itself. Do not write on the back of pages. If you lack space for an answer, then use the blank space on page 9. We will not grade answers written on scratch paper.

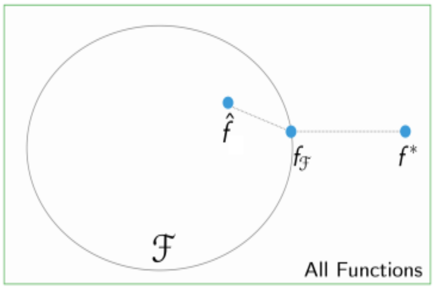
Name: ______________________________

NYU NetID: ______________________________

NYU Email: ______________________________
(as it appears on Gradescope)

Question	Points	Score
Decomposing Risk	11	
Regularization	8	
Scaling	8	
Gradient Descent	11	
Loss Functions	10	
Decision Boundaries	4	
Kernels	9	
SVM	12	
Total:	73	

1. Consider input space $\mathcal{X}$, output space $\mathcal{Y}$ and action space $\mathcal{A}$. Fix a loss function ℓ on $\mathcal{A} \times \mathcal{Y}$. Consider hypothesis space $\mathcal{F}$ of functions from $\mathcal{X}$ to $\mathcal{A}$. Fix a sample S drawn from $\mathcal{X} \times \mathcal{Y}$. Take



- $f^* = \underset{f}{\operatorname{argmin}}\, \mathbb{E}\left[\ell(f(x), y)\right]$
- $f_{\mathcal{F}} = \underset{f \in \mathcal{F}}{\operatorname{argmin}}\, \mathbb{E}\left[\ell(f(x), y)\right]$
- $\hat{f} = \underset{f \in \mathcal{F}}{\operatorname{argmin}}\, \frac{1}{m} \sum_{i=1}^{m} \ell(f(x_i), y_i)$

where m is the number of samples in S.

(a) Recall that the approximation error is the difference of risks $R(f_{\mathcal{F}}) - R(f^*)$.

i. (1 point) The approximation error is

[✓ **Positive or Zero**] ☐ Negative or Zero ☐ Cannot be Determined

ii. (1 point) The approximation error is

☐ Random [✓ **Non-Random**] ☐ Cannot be Determined

iii. (1 point) If we increase the size of $\mathcal{F}$, then the approximation error is

☐ Increased or Unchanged [✓ **Decreased or Unchanged**] ☐ Cannot be Determined

iv. (1 point) If we increase the size of S, then the approximation error is

☐ Changed [✓ **Unchanged**] ☐ Cannot be Determined

v. (1 point) Do we need to know the data generating distribution to compute the approximation error?

[✓ **True**] ☐ False

(b) Recall that the estimation error is the difference of risks $R(\hat{f}) - R(f_{\mathcal{F}})$.

i. (1 point) The estimation error is

[✓ **Positive or Zero**] ☐ Negative or Zero ☐ Cannot be Determined

ii. (1 point) For fixed sample S, the estimation error is

☐ Random [✓ **Non-Random**] ☐ Cannot be Determined

iii. (1 point) If we increase the size of $\mathcal{F}$, then the estimation error is

[✓ **Increased or Unchanged**] ☐ Decreased or Unchanged [✓ **Cannot be Determined**]

iv. (1 point) If we increase the size of S, then the estimation error is

[✓ **Changed**] ☐ Unchanged [✓ **Cannot be Determined**]

v. (1 point) Do we need to know the data generating distribution to compute approximation error

[✓ **True**] ☐ False

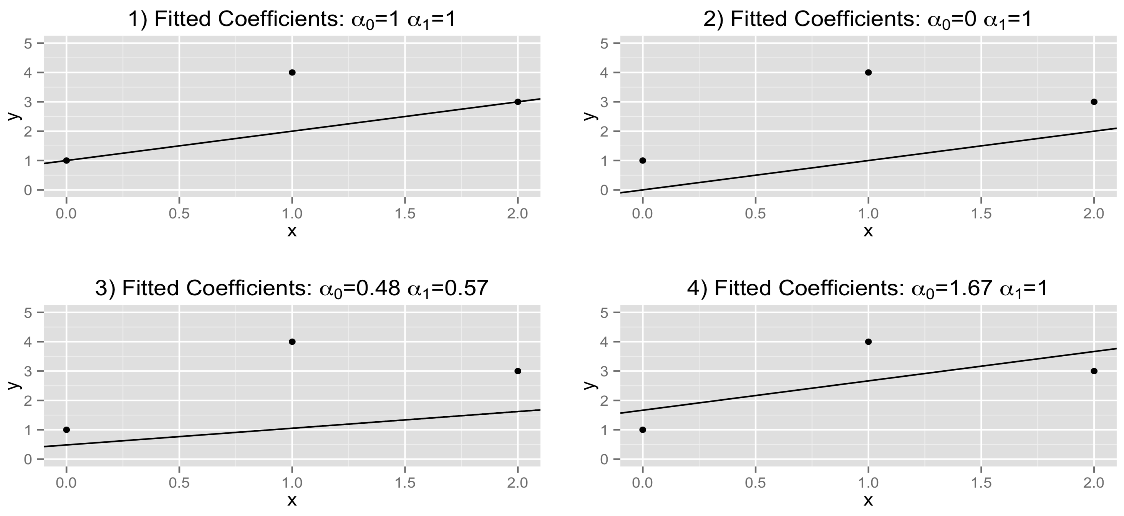
(c) (1 point) For some models like Lasso Regression, we have different approaches to fitting the training data. Each approach attempts to find $\widehat{f}$. Does the choice of the approach affect

☐ Approximation Error ☐ Estimation Error [✓ **Neither**]

2. (a) (8 points) We have a dataset $\mathcal{D} = \{(0,1), (1,4), (2,3)\}$ that we fit by minimizing an objective function of the form:

$$J(\alpha_0, \alpha_1) = \lambda_1 (\alpha_0 + \alpha_1) + \lambda_2(\alpha_0^2 + \alpha_1^2) + \sum_{i=1}^{3} (\alpha_0 + \alpha_1 x_i - y_i)^2 ,$$

and the corresponding fitted function is given by $f(x) = \alpha_0 + \alpha_1 x$. We tried four different settings of λ_1 and λ_2, and the results are shown below.



For each of the following parameter settings, give the number of the plot that shows the resulting fit.

i. ____**1**____ $\lambda_1 = 0$ and $\lambda_2 = 2$.

ii. ____**4**____ $\lambda_1 = 0$ and $\lambda_2 = 0$.

iii. ____**3**____ $\lambda_1 = 0$ and $\lambda_2 = 10$.

iv. ____**2**____ $\lambda_1 = 5$ and $\lambda_2 = 0$.

3. Suppose we have input space $\mathcal{X} = \{-1.5, -0.5, 0.5, 1.5\} \times \{-0.001, 0.001\}$, output space $\mathcal{Y} = \{-1, 1\}$ and action space $\mathbb{R}$. Assume the following about the data generating distribution

   - Y coordinate has equal probability of being $-1, 1$
   - X_1 coordinate has equal probability of being $\{-1.5, -0.5, 0.5, 1.5\}$. X_1 is related to Y through $X_1 = Y - 0.5Z$ where $Z = \pm 1$ with equal probability
   - X_2 has equal probability of being $\{-0.001, 0.001\}$. X_2 is related to Y through $X_2 = Y/1000$

   Suppose we have Ridge Regression with m samples

$$J(\mathbf{w}) = \lambda(w_1^2 + w_2^2) + \frac{1}{m}\sum_{i=1}^{m}\left(w_1 x_1^{(i)} + w_2 x_2^{(i)} - y_i\right)^2$$

   We're trying to decide between weights $\mathbf{w}_{\text{accurate}} = [0, 1000]$ and $\mathbf{w}_{\text{small}} = [1, 0]$.

   (a) (2 points) What is the value of $J(\mathbf{w}_{\text{accurate}})$?

   ☐ 1000λ ☐ 1000 ■ $1000^2\lambda$ ☐ 1000^2

   (b) (2 points) For large values of m, the empirical risk

$$\frac{1}{m}\sum_{i=1}^{m}\left(w_1 x_1^{(i)} + w_2 x_2^{(i)} - y_i\right)^2$$

   approximates the statistical risk

$$\mathbb{E}\left[(w_1 X_1 + w_2 X_2 - Y)^2\right] .$$

   Use the statistical risk to approximate the value $J(\mathbf{w}_{\text{small}})$.

   ☐ $0.5 + \lambda$ ■ $0.25 + \lambda$ ☐ $1 + \lambda$ ☐ $0.75 + \lambda$

   (c) (2 points) Using your answers above, determine λ^* such that we would choose $\mathbf{w}_{\text{small}}$ for any $\lambda > \lambda^*$.

   Solution: $\frac{0.25}{1000^2-1}$

   (d) (2 points) For most values of λ, we would choose $\mathbf{w}_{\text{small}}$. How could we transform the features to avoid choosing the less accurate weights?

   Solution: Scaling features – for example, min-max scaler

4. Momentum is a variation of gradient descent where we include the gradient at a previous iteration in the current iteration. The update rule is

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \alpha \frac{\partial L}{\partial \mathbf{w}}\left(\mathbf{w}^{(t)}\right) - \gamma \frac{\partial L}{\partial \mathbf{w}}\left(\mathbf{w}^{(t-1)}\right)$$

Here L is the objective function and $\alpha, \gamma > 0$ are the learning rates. Assume for iteration $t = 0$ and $t = -1$, we set $\mathbf{w}^{(t)} = w_0$ the initial guess.

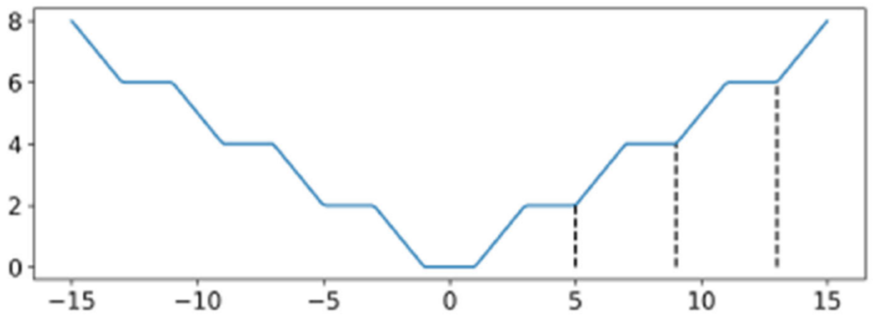


Figure 1: Graph of objective function L

(a) Refer to the chart in Figure 1.

i. (1 point) Assuming that $\mathbf{w}$ starts in a flat region that is not a minimum and $\alpha > 0$, will the basic gradient descent algorithm terminate at a minimum? Note that the basic gradient descent algorithm is the momentum gradient descent algorithm with $\gamma = 0$

☐ Yes with enough iterations ☐ Maybe [✓ **Never**]

ii. (1 point) Assuming that $\mathbf{w}$ starts in a sloped region and $\alpha > 0$, will the basic gradient descent algorithm terminate at a minimum?

☐ Yes with enough iterations [✓ **Maybe**] ☐ Never

iii. (1 point) Assuming that $\mathbf{w}$ starts in a flat region that is not a minimum and both $\alpha > 0$ and $\gamma > 0$, will the momentum gradient descent algorithm terminate at a minimum?

☐ Yes with enough iterations ☐ Maybe [✓ **Never**]

iv. (1 point) Assuming that $\mathbf{w}$ starts in a sloped region and both $\alpha > 0$ and $\gamma > 0$, will the momentum gradient descent algorithm terminate at a minimum?

☐ Yes with enough iterations [✓ **Maybe**] ☐ Never

v. (1 point) Is $L(\mathbf{w})$ convex?

☐ Yes [✓ **No**] ☐ No, but $-L(\mathbf{w})$ is convex ☐ No, but $L(-\mathbf{w})$ is convex

(b) (6 points) Fill in the twelve blanks in the code with the following variables to implement gradient descent with momentum.

w	X	y	w_prev	num_iter	w0
temp	alpha	gamma	range	len	t

Note that the same variable can be used multiple times. Some variables may not be used at all. Only use one variable per blank.

```python
def grad(X, y, w):
    ''' Returns gradient dL/dw at w
    X: matrix, training data features
    y: vector, training data labels
    w: vector, weights '''

def grad_desc_momentum(X, y, num_iter, alpha, gamma, w0):
    ''' Returns weights w computed after num_iter iterations.
    X: matrix, training data features
    y: vector, training data labels
    num_iter: number, number of iterations to run
    alpha: number, learning rate
    gamma: number, learning rate for momentum
    w0: weights for t=0 and t=-1 '''

    w, w_prev = ______<i>_______, ______<ii>______
    for ___<iii>_____ in ____<iv>____(_________<v>_______):
        g = grad(X, y, w)
        m = grad(X, y, ______<vi>_____)
        __<vii>___, ___<viii>___ = ___<ix>____ - ___<x>___ * g \
                                      - __<xi>___ * m, ____<xii>___
    return w
```

i.	w0	v.	num_iter	ix.	w
ii.	w0	vi.	w_prev	x.	alpha
iii.	t	vii.	w	xi.	gamma
iv.	range	viii.	w_prev	xii.	w

5. Consider input space $\mathcal{X} = \{1, 2, 3, 4\}$, output space $\mathcal{Y} = \{1, 2, 3, 4\}$ and action space $\mathbb{R}$. Take the square loss: $\ell(\hat{y}, y) = (\hat{y} - y)^2$.

   (a) (3 points) Fix x. Determine the constant c such that $\mathbb{E}\left[(Y - c)^2 | X = x\right]$ is minimized. Note that you need to take a derivative.

   Solution: $c = E[Y|X]$

   (b) (3 points) Assume the following about the data generating distribution

   - The coordinate X is uniformly distributed on $\mathcal{X}$. So equal probability $\frac{1}{4}$ to features $\{1, 2, 3, 4\}$.

- The coordinate Y given the coordinate X is uniformly distributed on $\{1, \ldots, x\}$. So equal probabilities $\frac{1}{x}$ to labels $\{1, \ldots, x\}$ conditional on feature x.

What is the target function? In other words, for fixed x how should we choose $f^*(x)$ to minimize the expected square loss.

Solution: $f^*(x) = (x+1)/2$.

(c) (4 points) What is the expected square loss of the target function?

Solution:

$$E[(Y - f^*(X))^2] = E[E[(Y - (X+1)/2)^2 | X]] = \frac{26}{48}.$$

6. (a) (2 points) Figure 2 contains a training set $\{x_1, x_2, \ldots, x_{25}\}$. Below we have several feature transformations. By themselves, which might allow us to separate the transformed data with a linear decision boundary? Select **all** possible choices.

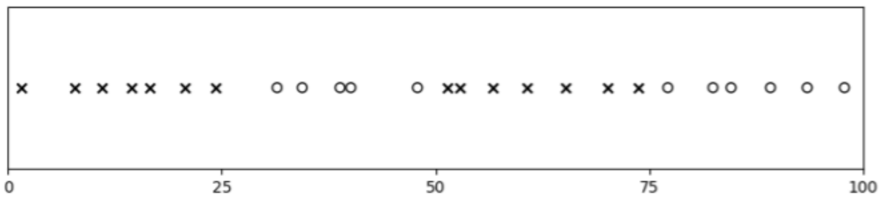


Figure 2: Training Data

☐ Centering the data

☐ Add a feature x^2

■ **Add a feature that is 1 if $x \leq 50$ or -1 if $x > 50$**

■ **Add two features x^2 and x^3**

(b) (2 points) Figure 3 contains a training set $\{(x_1^{(1)}, x_2^{(1)}), \ldots, (x_1^{(100)}, x_2^{(100)})\}$. Below we have several feature transformations. By themselves, which might allow us to separate the transformed data with a linear decision boundary? Select **all** possible choices.

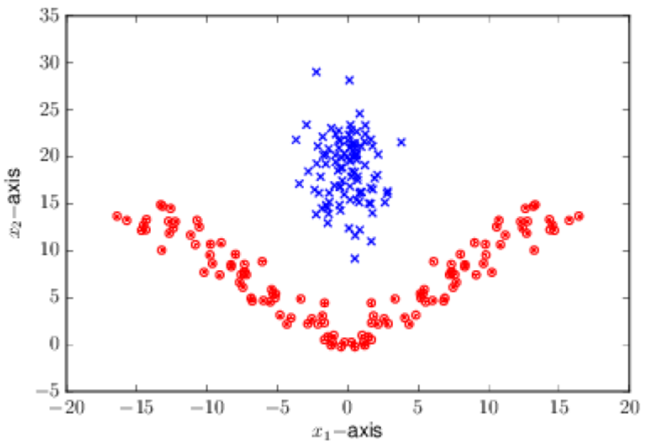


Figure 3: Training Data

☐ Scaling the data

■ **Adding features x_1^2, x_2^2, x_1x_2**

☐ Adding a feature that is 1 if $x_2 \geq 10$ or -1 if $x_2 < 10$

■ **Adding a feature $|x_1|$**

7. Define the Huber loss function $h : \mathbb{R} \to \mathbb{R}$ by

$$h(x) = \begin{cases} x^2/2 & \text{if } |x| \leq 1, \\ |x| - 1/2 & \text{if } |x| > 1. \end{cases}$$

Consider the objective function

$$J(w) = \lambda \|w\|_2^2 + \frac{1}{n} \sum_{i=1}^{n} h(w^T x_i - y_i)$$

where $(x_1, y_1), \ldots, (x_n, y_n) \in \mathbb{R}^d \times \mathbb{R}$. Fix $\lambda > 0$. Note that the function is differentiable.

(a) (3 points) We want to minimize $J(w)$ using stochastic gradient descent. Assume the current data point is (x_i, y_i). The step direction is given by $v = -\nabla_w G(w)$, for some function $G(w)$. Give an explicit expression for $G(w)$ in terms of h, λ, and the given data. You do not have to expand the function h.

Solution:

$$G(w) = h(w^T x_i - y_i) + \lambda \|w\|_2^2$$

(b) (3 points) Assume $J(w)$ has a minimizer w^*. Give an expression for w^* in terms of a vector $\alpha \in \mathbb{R}^n$ that is guaranteed by the representer theorem. You may use the design matrix $X \in \mathbb{R}^{n \times d}$.

Solution:

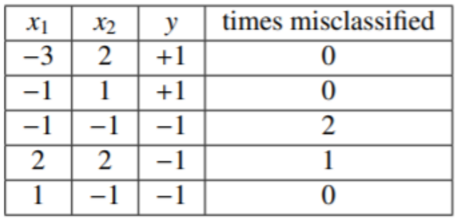
$$w^* = \sum_{i=1}^{n} \alpha_i x_i = X^T \alpha$$

(c) (3 points) Let $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be a Mercer kernel, and let $K \in \mathbb{R}^{n \times n}$ denote the Gram matrix $K_{ij} = k(x_i, x_j)$. Give a kernelized form of the objective J in terms of K. Recall that $w^T x_i = (Xw)_i$ where $X \in \mathbb{R}^{n \times d}$ is the matrix with ith row x_i^T.

Solution:

$$J(\alpha) = \frac{1}{n} \sum_{i=1}^{n} h((K\alpha)_i - y_i) + \lambda \alpha^T K \alpha.$$

8. Figure 4 shows a training set in $\mathbb{R}^2$. Suppose that we use perceptron algorithm for classification. We record the total number of times each point occurs in the update step. Remember that if a point is misclassified, then it occurs in the update step.



x_1	x_2	y	times misclassified
-3	2	$+1$	0
-1	1	$+1$	0
-1	-1	-1	2
2	2	-1	1
1	-1	-1	0

Figure 4: Training Data

(a) i. (3 points) Assume that the initial weight is $w^{(0)} = [-3, 2, 1]$ where 1 is the offset term. So the feature $x_3 = 1$ is constant.
What is the equation of the separating line expressed in terms of x_1 and x_2 determined by the algorithm ?

Solution: $-3x_1 + 2x_2 - 2 = 0$

ii. (1 point) In some cases, removing a single point can change the decision boundary. Here would removing a single point from the training set change the decision boundary? Please explain your answer.

Solution: The two points for the update.

iii. (2 points) If we added the point $[2, -2]$ with label $+1$ to the training set, then would we obtain different results? In particular, would the algorithm converge?

Solution: Data would not be linearly separable so algorithm would not converge

Figure 5 shows training data with two classes. We want to use hard-margin support vector machine. Remember that we choose the decision boundary to maximize the margin.

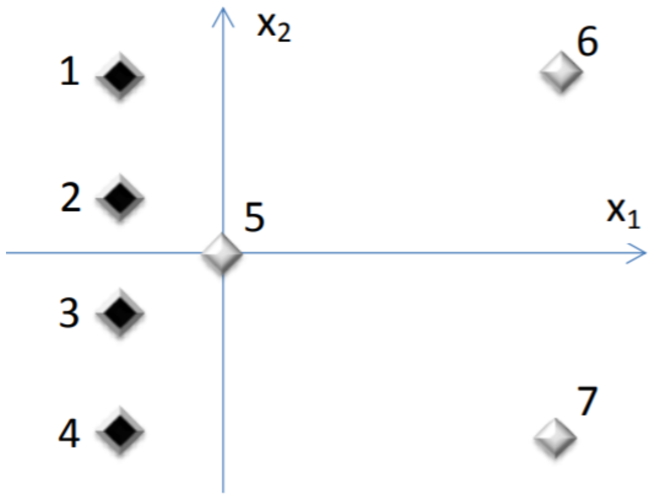


Figure 5: Training Data

(b) i. (2 points) Draw the decision boundary obtained by the hard-margin SVM method with a solid line. Draw the margins on either side with dashed lines.

ii. (1 point) What are the **possible** support vectors. Please indicate the number.

Solution: 1,2,3,4,5

iii. (1 point) What is the classification error on the training set? In other words, how many points are incorrectly classified?

Solution: 0

iv. (1 point) Would the removal of a single point change the decision boundary? If so, then what points?

Solution: Point 5

v. (1 point) Suppose we use leave-one-out cross validation meaning we use 7-fold cross validation with a split of 6 to 1 between training set and validating set. Compute the average classification error over the 7-folds.

Solution: 1/7