CSCI 4100/6100 RPI
Machine Learning From Data 2017

FINAL: 90 Minutes

Last Name: Solutions

First Name: ________________

RIN: ________________

Section: 4100 / 6100 (circle one)

Answer **ALL** questions.
NO COLLABORATION or electronic devices. Any violations result in an F.
NO questions allowed during the test. Interpret and do the best you can.
ALWAYS show your work and justify each answer.

GOOD LUCK!

1	2	3	4	5	6	7	8	9	10	11	TOTAL
/1	/1	/1	/1	/1	/1	/1	/1	/2	/4	/1	/15

- **You do not have time to waffle.**
- **Keep your answers precise and concise.**
- **Questions (or parts) are graded** 0%, 50%, 100%.

1. What is the definition of *learning*? What is the 2-step approach to learning and why do it that way?

Learning: Given data, output $g(x)$ which approximates $f(x)$ (low E_{out})

2-steps ① Ensure $E_{in} \approx E_{out}$ ② Ensure E_{in} is small

Why: We cannot measure E_{out}
We can only measure E_{in} so we minimize E_{in} (step 2)
To learn we want E_{out} small which we must then link to E_{in} (step 1)

2. Every card has a letter on one side and a number on the other.
Hypothesis: If a card has a P on it, then the other side is a 5.

$\boxed{S}\ \boxed{5}\ \boxed{P}\ \boxed{T}\ \boxed{3}\ \boxed{4}$

Above are some cards which you may turn (data). Can the hypothesis be falsified by the data. If not, why? If so, which cards are the fewest you need to turn over to ~~generate evidence for the hypothesis~~ see if the hypothesis is false?

Any P: must check other side is 5 $\nearrow$ Yes → evidence
$\searrow$ No → falsified.

Any non-5: must check other side is NOT P $\nearrow$ NOT P → evidence
$\searrow$ P → falsified.

Minimum # cards to turn to see if the hypothesis is false: $\boxed{P}$, $\boxed{3}$, $\boxed{4}$ (3 cards)

3. The professor of a class released the previous final just one day before the final (hence only giving students a limited time to study the previous final). Why? *[Hints: The professor wanted to prevent the students from doing what? The professor wanted the students to use the previous final as a what?]*

Prevent students from overfitting to the final questions.

Wanted students to use the previous final as a test set to test how well student learning generalized to out of sample.

4. You logged into facebook and saw all the great things your friends are posting. This got you depressed when you compared those activities with your life. What learning from data trap did you fall into?

Sampling Bias!

People post the good things.
Don't compare this with the "random" things in your life.

5. When we studied the VC-theory of generalization, we identified two types of hypothesis set, good and bad. What is the difference between the two and why is this difference important?

good → finite VC-dimension
bad → infinite VC - dimension

$E_{out} \leq E_{in} + \Omega(\mathcal{H})$

finite → $\Omega(\mathcal{H}) \in O\left(\sqrt{\frac{d_{VC}}{N} \ln N}\right) \to 0$ when $N \to \infty$

→ 1'st step of learning. ($E_{in} \approx E_{out}$) is ensured.

6. With 10,000 data points, you used 8,000 for training and kept 2,000 as a validation set to determine a good regularization parameter, which turned out to be $\lambda = 0.08$. You now thought of two strategies.

S1: Output the final g trained on the 8,000 training points using $\lambda = 0.08$ for regularization.
S2: Output the final g trained on all the 10,000 data points using $\lambda = 0.08$ for regularization.

Discuss the pros and cons of each strategy. All things considered, which one do you go with?

S1: According to validation you output the best g from training with 8000 points.
Con: Didn't use all data to "train" g with $\lambda = 0.08$

S2: Pro: used all data to train g with $\lambda = 0.08$:
$\lambda = 0.08$ may be too much regularization with 10000 points.

I would go with the more data since over-regularizing a little is generally OK.

7. "We are regularized by our parents." Explain.

We are learning machines in a noisy environment. Without constraint (parents) we would overfit and do only things which minimize E_{in} (ie. how we feel).

<u>Eg:</u> the child will always eat candy!

8. What is overfitting? What are the causes? What are the tools to fight it?

Overfitting: fitting the data <u>more</u> than you should: $E_{in} \downarrow$ $E_{out} \uparrow$.

<u>Causes</u> : Deterministic & Stochastic Noise.

<u>Tools</u> : Regularization ↓ constrain the model   Validation ↓ Estimate E_{out}.

9. Using the nearest neighbor rule, and test point $\mathbf{x}$, assume $\pi(\mathbf{x}) = \pi(\mathbf{x}_{[1]})$ (recall $\pi(\mathbf{x})$ is the probability $y(\mathbf{x}) = +1$). *Prove:* $E_{\text{out}} \leq 2E^*(1 - E^*)$, where, for test point $\mathbf{x}$, E^* is *optimal* probability of misclassification and E_{out} is the probability of misclassification by the nearest neighbor rule.

$$P[\text{misclassify}] = P[\text{misclassify} \mid x_{[1]} = 1]\, P[x_{[1]} = 1] + P[\text{misclassify} \mid x_{[1]} = 0]\, P[x_{[1]} = 0]$$

$$= (1 - \pi(x))\, \underset{\substack{\parallel \\ \pi(x)}}{\pi(x_{[1]})} + \pi(x)\, (1 - \underset{\substack{\parallel \\ \pi(x)}}{\pi(x_{[1]})}).$$

$$= 2\pi(x)(1 - \pi(x)).$$

$$E^* = \min(\pi(x), 1 - \pi(x))$$

$$E^*(1 - E^*) = \pi(x)(1 - \pi(x))$$

$$\rightarrow \quad P[\text{misclassify}] = E_{out} = 2E^*(1 - E^*)$$

10. Give formulas for the form of the final *classification* hypothesis in each case. Specify which parameters (and their dimension) are to be learned from data. The data are $(\mathbf{x}_n, y_n)$, where $\mathbf{x}_n \in \mathbb{R}^d$ and $y_n = \pm 1$.

(a) Linear model trained with the pocket algorithm.

$g(x) = \text{sign}(w^T x)$ learn $w \in R^{d+1}$

(b) Linear model with feature transform $\mathbf{z} = \mathbf{x}^{\mathrm{T}}\mathbf{x}$, trained with the pocket algorithm.

$g(x) = \text{sign}(w_0 + w_1 x^T x)$ learn w_0, w_1

(c) Nearest neighbor.

$g(x) = y_{[1]}$

$y_{[1]}$ is the y for the nearest-neighbor

NO PARAMETERS to learn.

(d) Nonparametric RBF.

$$g(x) = \text{sign}\left(\frac{\sum_{n=1}^{N} y_n \, \phi\left(\frac{\|x - x_n\|}{r}\right)}{\sum_{m=1}^{N} \phi\left(\frac{\|x - x_m\|}{r}\right)} \right)$$

$\phi(z) = e^{-\frac{z^2}{2}}$ (for example (Gaussin Kernel))

NO PARAMETERS to learn.

(e) k-RBF network.

$$g(x) = \text{sign}\left(\sum_{j=1}^{K} w_j \, \phi\left(\frac{\|x - \mu_j\|}{r}\right) + w_0 \right)$$

learn $w_0, \underbrace{w_j, \mu_j}_{j=1\dots K} \in R^d$.

(f) Two hidden-layer neural network with function $\theta(\cdot)$ in each hidden unit.

1 hidden layer →

$$g_\alpha(x) = \text{sign}\left(w_0^\alpha + \sum_{j=1}^{H_1} w_j^\alpha \, \theta(v_j^T x) \right)$$

learn w_0^α, w_j^α $\alpha = 1 \cdots H_2$, $j = 1 \cdots H_1$

$v_j \in R^{d+1}$

$u_0 \dots u_{H_2}$

2 hidden Layer: $g(x) = \text{sign}\left[u_0 + \sum_{\alpha=1}^{H_2} u_\alpha \, \theta\left[w_0^\alpha + \sum_{j=1}^{H_1} w_j^\alpha \theta(v_j^T x) \right] \right]$

(g) SVM with feature transform $\mathbf{z} = \phi(\mathbf{x})$.

$g(x) = \text{sign}(\tilde{w}^T \phi(x) + \tilde{b})$ learn $\tilde{w}, \tilde{b}$

(h) SVM with kernel $K(\mathbf{x}, \mathbf{y})$.

$$g(x) = \text{sign}\left(\sum_{n=1}^{N} \alpha_n^* y_n K(x_n, x) + b^* \right)$$

learn α_n^* b^*

11. How do your formulas above change if you are doing logistic regression instead of classification?

Replace sign with logistic sigmoid.

For Nearest neighbor ⇐ not making real sense.

for K-NN, use fraction of +1 in K-neighborhood as probability.