Course: STAT W4400
Title: Statistical Machine Learning
Semester: Spring 2015
Instructor: John P. Cunningham

MIDTERM EXAM

Do not open this exam until instructed. Carefully read the following instructions.

This exam is to be done in-class. You have 75 minutes to complete the entirety. Write your name, UNI, and the course title on the cover of the blue book. All solutions should be written in the accompanying blue book. No other paper (including this exam sheet) will be graded. **To receive credit for this exam, you must submit blue book with the exam paper placed inside.** As reference you may use one sheet of 8.5×11in paper, on which any notes can be written (front and back). No other materials are allowed (including calculators, textbooks, computers, and other electronics). To receive full credit on multi-point problems, you must thoroughly explain how you arrived at your solutions. Each problem is divided up into several parts. Many parts can be answered independently, so if you are stuck on a particular part, you may wish to skip that part and return to it later. Good luck.

1. **Short Answers** (20 points)

   (a) (3 points) All ensemble methods are built atop underlying weak learners that are trained and evaluated. What additional requirement does boosting place on the underlying weak learners?

   (b) (1 point) Does the random forest method use bagging or boosting?

   (c) (3 points) Can regularization reduce training MSE?

   (d) (3 points) Can regularization reduce testing MSE?

   (e) (3 points) Consider the convex optimization problem over $x \in \mathbb{R}^2$:

   $$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & g(x) = 0 \end{aligned}$$

   Consider a point x^* with $\nabla f(x^*) = \begin{bmatrix} 1.2 \\ 0.7 \end{bmatrix}$ and $\nabla g(x^*) = \begin{bmatrix} 3.6 \\ 2.1 \end{bmatrix}$. Is x^* a minimum?

   (f) (3 points) Consider the convex optimization problem over $x \in \mathbb{R}^2$:

   $$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & g(x) \leq 0 \end{aligned}$$

   Consider a point x^* with $\nabla f(x^*) = \begin{bmatrix} 1.2 \\ 0.7 \end{bmatrix}$ and $\nabla g(x^*) = \begin{bmatrix} 3.6 \\ 2.1 \end{bmatrix}$. Is x^* a minimum?

   (g) (2 points) What is one advantage of LASSO over ridge regression?

   (h) (2 points) What is one advantage of ridge regression over LASSO?

2. **Understanding Misclassification** (40 points)

The misclassification rate of machine learning classifiers depends substantially on the distribution of the data, the classification technique being used, and more. This question probes this dependency. In all problems below we will consider a binary classifier: $y_i \in \{-1, +1\}$.

(a) (3 points) If we are exclusively interested in minimizing misclassification rate, what loss function should we use?

(b) (5 points) Suppose that we are interested in the classification of an extremely rare event. It is usually the case in these settings that false positives are far less of a concern than false negatives. Would minimizing the misclassification rate still be an appropriate goal? Why or why not? If it is not appropriate, describe how you would modify the empirical risk function to better achieve our goals.

(c) (5 points) Now assume we know the class conditional distributions of the data: $p(\mathbf{x}|y = +1)$ is exactly spherical Gaussians with mean vector μ_+ and covariance $\sigma^2 I$, and similar for the -1 class: $p(\mathbf{x}|y = -1) = \mathcal{N}(x; \mu_-, \sigma^2 I)$. Notice that σ is the same for both classes, but the mean vectors are not. Describe what happens to the misclassification rate for different values of μ_+ , μ_- , σ^2. More specifically under what scenario would a classifier achieve a low misclassification rate, and conversely a high misclassification rate? Justify your answer.

(d) (5 points) If the data is distributed according to the above assumptions, what method should we use to learn the Bayes-optimal classifier?

(e) (12 points) Consider the following data set with covariates $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$: $x_1 \in \{True, False\}$, $x_2 \in \{Red, Green, Blue\}$, and class labels, $y \in \{-1, +1\}$:

y	x_1	x_2
-1	False	Green
-1	True	Green
-1	False	Blue
-1	True	Red
$+1$	False	Green
$+1$	False	Green
$+1$	False	Green
$+1$	True	Blue

Train the Naive Bayes classifier on this data: estimate all necessary probability distributions.

(f) (10 points) Classify the training data and calculate the misclassification rate of this classifier on the training data.

3. **Nearest Neighbor Regression** (40 points)

Consider a regression problem with training data $X = \{(\tilde{\mathbf{x}}_1, \tilde{y}_1), ..., (\tilde{\mathbf{x}}_n, \tilde{y}_n)\}$, for $\tilde{\mathbf{x}} \in \mathbb{R}^d$ and $y \in \mathbb{R}$. Given an integer parameter k and some data X', a k-nearest neighbor (kNN) regression is the function $f(\mathbf{x}; k, X') : \mathbb{R}^d \to \mathbb{R}$ that is computed by:

1. Finding the k points in X' closest to $\mathbf{x}$. That is, define $\eta_k(\mathbf{x}; X')$ as the function that returns the indices i corresponding to these closest points in terms of Euclidean distance $d(\mathbf{x}, \tilde{\mathbf{x}}_i) = ||\mathbf{x} - \tilde{\mathbf{x}}_i||_2$.
2. Assigning $f(\mathbf{x}; k, X')$ as the average of those k nearest neighbors.

- Mathematically, the above two operations are written as:

$$f(\mathbf{x}; k, X') = \frac{1}{k} \sum_{i \in \eta_k(\mathbf{x}; X')} y_i$$

- For example, consider the data X' and a query point $\mathbf{x} = 2.1$:

$\tilde{\mathbf{x}}_i$	2.0	1.4	1.5	2.5	3.6	1.2	1.9
y_i	11.3	12.1	10.1	14.1	10.1	19.3	10.6

Given $k = 3$, $\mathbf{x}$ has k nearest neighbors $\{\tilde{\mathbf{x}}_1 = 2.0,\ \tilde{\mathbf{x}}_4 = 2.5,\ \tilde{\mathbf{x}}_7 = 1.9\}$ (the function $\eta_k(\mathbf{x}; X') = \{1, 4, 7\}$). The regression $f(\mathbf{x}; 3, X')$ is then the average of $\{\tilde{y}_1 = 11.3,\ \tilde{y}_4 = 14.1,\ \tilde{y}_7 = 10.6\}$, namely $f(\mathbf{x}; 3, X') = 12.0$.

Notice that, for a given choice of k, the kNN classifier requires no training; each classification result is computed directly from X'.

(a) (5 points) Using the above data X', what is $f(\mathbf{x}; 2, X')$ for $\mathbf{x} = 1.4$?

(b) Describe a procedure to choose an optimal value for the parameter k out of the values $\{1, 3, 5, 7, 9\}$, using some data X Hint: note that k in some way controls the complexity of the regression function. In detail:

i. (5 points) Specify an empirical risk function you will use to compare different versions of the regression function. Give a formula for this quantity when evaluated on a subset X_{val} of X.

ii. (10 points) Write down a step-by-step procedure to choose k.

iii. (3 points) k-nearest neighbor regression often works surprisingly well. Can you think of a reason why this approach may nonetheless be a bad choice for an application running, for example, on a phone or a digital camera?

(c) Since kNN only computes distances between points, your next task is to exploit that fact to make a kernel version of this classifier, using the same kernel trick that we did in SVM. You may now assume k is given (or determined from your previous result). To derive *kernel* kNN, you must:

i. (3 points) Consider the squared Euclidean distance between any two points $\mathbf{x}$ and $\mathbf{x}'$. Expand the expression $d^2(\mathbf{x}, \mathbf{x}') = ||\mathbf{x} - \mathbf{x}'||_2^2$ in terms of vector inner products.

ii. (8 points) Assume you are given a kernel $K(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathcal{F}} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. Use the kernel trick to make the squared Euclidean distance from the previous part into a kernel squared distance. Write down this expression and call it $d_K^2(\mathbf{x}, \mathbf{x}')$.

iii. (3 points) d_K is itself a distance measure. What distance does it calculate?

iv. (3 points) Algorithmically, which operations in kNN are affected by this kernelized version of the method: the form of the function $f(\mathbf{x}; k, X')$, the form of the function $\eta_k(\mathbf{x}, X')$, neither, or both?