CS4780 Final

Spring 2017

NAME:	
Net ID:	
Email:	

1 [??] General Machine Learning

Please identify if these statements are either True or False. Please justify your answer **if false**. Correct "True" questions yield 1 point. Correct "False" questions yield two points, one for the answer and one for the justification.

1. (**T/F**) Random forests is one of the few machine learning algorithms that makes no assumptions on the data.

2. (**T/F**) One implication of the curse of dimensionality is that if you sample n data points uniformly at random within a hyper cube of dimensionality d, all pairwise distances converge to 0 as $n \to \infty$.

3. During training, in a linearly separable data set, the perceptron algorithm never misclassifies the same input twice. (**T/F**)

4. (**T/F**) You have a biased coin and toss it n times. The MAP estimate with +1 smoothing of the probability of getting "head" is $\frac{n_H+1}{n+1}$, where n_H is the number of occurrences of "head" amongst your n throws.

5. (**T/F**) The multinomial Naive Bayes algorithm is a linear classifier.

6. (**T/F**) MAP inference maximizes $P(\mathbf{w}|Data)$ whereas MLE maximizes $P(Data; \mathbf{w})$, where $\mathbf{w}$ represents the model parameters.

7. (**T/F**) Newton's Method diverges only if the Hessian matrix is not invertible.

8. (**T/F**) Linear (ordinary least squares) regression can be solved in closed form, although sometimes that is computationally impractical or even infeasible.

9. (**T/F**) SVMs maximize the margin between the training and testing data.

10. (**T/F**) In order for gradient descent to converge, the loss function has to be convex and differentiable everywhere.

11. (**T/F**) The bias variance trade-off decomposes the error obtained by a classifier into (squared) bias, variance, and noise. The noise term cannot possibly be addressed, even by changing the feature representation of the data.

12. (**T/F**) In a setting of high bias, a great remedy is to add more training data.

13. (**T/F**) Bagging reduces variance.

14. (**T/F**) Boosting reduces noise.

15. (**T/F**) Learning with kernels is expensive, because the data is mapped into a very high dimensional space and therefore storing the transformed data consumes a lot of storage.

16. (**T/F**) The mean prediction of Gaussian processes is identical to kernelized linear regression.

17. (**T/F**) One popular application of Gaussian Processes is to find hyper-parameters of machine learning algorithms.

18. (**T/F**) Ball-Trees are a data structure to speed up the perceptron algorithm.

19. (**T/F**) Decision Trees stop splitting when the impurity function can no longer be improved with a single split.

20. (**T/F**) Random Forests are bagged decision trees with one additional modification: Each splitting dimension is chosen completely uniformly at random.

21. (**T/F**) Provided each weak learner can classify a weighted version of the training data set with better than 0.5 accuracy, in AdaBoost the training error reduces exponentially.

22. (**T/F**) Deep neural networks are great on many data sets, but do not work competitively on image classification tasks.

23. (**T/F**) The optimization of deep neural networks is a convex minimization problem.

2 [21] Bias Variance / Model Selection

1. (3) Write down the bias (squared), variance, noise decomposition of the expected test error $\mathbb{E}_{x,y,D}\left[(h_D(x)-y)^2\right]$.

2. (5) Describe how to detect settings with high bias and provide three approaches that could help reduce the bias.

3. (13) For each of the following scenarios, determine if the model has low/high bias and variance. Explain your choice.

   (a) Logistic regression on *linearly* separable data and *non-linearly* separable data.

   (b) kNN with small k and large k.

   (c) Uniform random labeling.

3 [21] Kernel Methods

1. Suppose you are using a kernel SVM with the RBF kernel $k(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{z}\|_2^2}{\sigma^2}\right)$ to do classification. Recall that the kernel SVM is trained by solving the dual optimization problem:

$$\min_{\alpha_1,\ldots,\alpha_n} \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{K}_{ij} - \sum_i \alpha_i$$
$$\text{s.t. } 0 \leq \alpha_i \leq C$$
$$\sum_i \alpha_i y_i = 0$$

   (a) (5) Assume you can either set C and σ^2 to a very large value ($\gg 0$) or a very small value (ϵ). Provide a setting with high bias and one with high variance. *Briefly* explain your answers.

   (b) (3) $\mathbf{x}_1$ turns out to be a support vector. What can you say about its corresponding optimal value α_i^* and the margin between the hyperplane and $\mathbf{x}_1$?

   (c) (5) In order to apply the classifier to a test point, we need the hyper-plane bias b. Show how b can be recovered from $\alpha_1^*, \ldots, \alpha_n^*$ with the help of the support vector $\mathbf{x}_i$ and label $y_i \in \{-1. +1\}$.

2. (8) For this question, you will find the following rules about recursively building kernels helpful. Given kernels $k_1(\mathbf{x}, \mathbf{z})$ and $k_2(\mathbf{x}, \mathbf{z})$, the following are well-defined kernels:

$$k(\mathbf{x}, \mathbf{z}) = \mathbf{x}^\top A\mathbf{z}, A \succeq 0 \quad (1)$$
$$k(\mathbf{x}, \mathbf{z}) = ck_1(\mathbf{x}, \mathbf{z}) \quad (2)$$
$$k(\mathbf{x}, \mathbf{z}) = \exp\left(k_1(\mathbf{x}, \mathbf{z})\right) \quad (3)$$
$$k(\mathbf{x}, \mathbf{z}) = f(\mathbf{x})k_1(\mathbf{x}, \mathbf{z})f(\mathbf{z}) \quad (4)$$
$$k(\mathbf{x}, \mathbf{z}) = k_1(\mathbf{x}, \mathbf{z}) + k_2(\mathbf{x}, \mathbf{z}) \quad (5)$$

Suppose that $\mathbf{x}, \mathbf{z} \in \mathbb{R}^2$. Let $[\mathbf{x}]_1$ and $[\mathbf{x}]_2$ denote the first and second coordinates of $\mathbf{x}$, respectively. Show that

$$k(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{\|[\mathbf{x}]_1 - [\mathbf{z}]_1\|_2^2}{\sigma^2}\right) + \exp\left(-\frac{\|[\mathbf{x}]_2 - [\mathbf{z}]_2\|_2^2}{\sigma^2}\right)$$

is a kernel.
Hint: You may find the following two matrices helpful:

$$A_1 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

You can assume they are positive semi-definite (i.e. $A_1 \succeq 0, A_2 \succeq 0$).

4 [21] CART

1. (2) Imagine you build a KD-Tree and label each leaf with the most common label amongst all training points that fall into this leaf. Why would this not be a desirable classifier?

2. (4) Name two reasons why Random Forests are such popular classifiers amongst practitioners?

3. (2) Assume you pre-process all your features in the following way: you sort each feature independently. For each feature, you then assign all those inputs that share the lowest feature value a new feature value of 1, all those with the second lowest value a 2, etc. How does this affect the trees that you construct?

4. (4) Under what conditions on your training set will a CART tree (with unlimited depth) obtain 0% training error.

5. (3) You are building a regression tree with the squared loss impurity. i.e. the labels in the leaf are $L = \{y_1, \ldots, y_m\}$ and the loss, under prediction t, is $\sum_{y \in L}(y - t)^2$. Prove that the average label $t = \frac{1}{m}\sum_{i=1}^{m} y_i$ minimizes the loss at a leaf.

6. (6) You are now considering minimizing the absolute loss instead: $\sum_{y \in L} |y - t|$. Define $L_{\leq} = \{y \in L : y \leq t\}$ and $L_{>} = \{y \in L : y > t\}$. Prove that setting t to the median of L minimizes this loss. To simplify things you can assume you have an odd number of samples (i.e. $m = 2r + 1$) and that all $y_i \in L$ are distinct (i.e. $y_i \neq y_j$ for any $y_i, y_j \in L$). (Without loss of generality it is sufficient to show there is no better splitting value t' that is *larger* than the median.)

5 [21] Boosting / Bagging

1. (3) Name two algorithms, for which boosting will be ineffective. Briefly justify why.

2. (5) Describe what happens in AdaBoost if two training inputs (in a binary classification problem) are identical in features but have different labels?

3. (3) In neural networks bagging can be performed *without random subsampling of the data*. i.e., one trains m neural networks independently and ensembles their results. Can you explain why the subsampling is unnecessary in this case?

4. Assume you have weak learners $h \in \mathcal{H}$ s.t. $h(\mathbf{x}) \in \{+1, -1\}$ for any $\mathbf{x}$. You are trying to apply boosting with the logistic loss function

$$\mathcal{L}(H) = \sum_{i=1}^{n} \ln\left(1 + e^{-y_i H(\mathbf{x}_i)}\right). \tag{6}$$

(remember, ln here refers to the natural logarithm)

(a) (4) Compute the derivative $\frac{\partial \mathcal{L}(H)}{\partial H(\mathbf{x}_i)}$.

(b) (6) Let $w_i = \frac{1}{1+e^{y_i H(\mathbf{x}_i)}}$ and let $\epsilon(h) = \sum_{i:h(\mathbf{x}_i)\neq y_i} w_i$ be the weighted error of the training set. For simplicity assume we are using a fixed step-size of 1. Show that the next classifier to be added to the ensemble H in order to minimize the loss function is $h = \text{argmin}_h \mathcal{L}(H + h) = \arg\min_h \epsilon(h)$.

6 [21] Deep Learning

Assume you are given a neural network with L layers to minimize a loss function $\mathcal{L}$

$$\begin{aligned} h(\mathbf{x}) =& \mathbf{w}^\top \phi_1(\mathbf{x}) && (7)\\ \phi_1(\mathbf{x}) =& \sigma(\mathbf{U}_1 \phi_2(\mathbf{x})) && (8)\\ &\vdots && (9)\\ \phi_\ell(\mathbf{x}) =& \sigma(\mathbf{U}_\ell \phi_{\ell+1}(\mathbf{x})) && (10)\\ &\vdots && (11)\\ \phi_L(\mathbf{x}) =& \sigma(\mathbf{U}_L \mathbf{x}) && (12) \end{aligned}$$

(*Note that the subscript of ϕ starts at 1 at the end of the network, and increases to L as we make our way back to the start*)

1. (5) Let us define $a_\ell = \mathbf{U}_\ell \phi_{\ell+1}(\mathbf{x})$ such that $\phi_\ell = \sigma(a_\ell)$. Let $\delta_\ell = \frac{\partial \mathcal{L}}{\partial a_\ell}$. Express $\frac{\partial \mathcal{L}}{\partial \mathbf{U}_\ell}$ in terms of δ_ℓ. (assume $1 < \ell < L$)

2. (5) Assume that the derivative of $\sigma(z)$ is given as $\sigma'(z)$. Define $\delta_{\ell+1}$ as a function of δ_ℓ. (assume $1 < \ell < L$) where $x = \phi_{L+1}$.

3. (3) Provide one reason why stochastic gradient descent can be better than traditional (batch) gradient descent when applied to neural networks.

4. (4) Assume you make all transition functions the identity (i.e. $\sigma(z) = z$). Prove that the final classifier is simply a linear classifier of the form $h(\mathbf{x}) = \hat{\mathbf{w}}^\top \mathbf{x}$ for some vector $\hat{\mathbf{w}}$.

5. (4) ML-practitioners tend to drop the learning rate during training. Explain why and what effect it has.

This page is left blank for scratch space.

This page is left blank for jokes. (Nothing too dirty.)

Please do not write on this page. For administrative purposes only.

T/F	
BV	
Kernels	
CART	
ENSEMBLE	
DL	
TOTAL	