COMPSCI 671 Exam 1

12th February 2019

Total Time: (8:30 AM to 9:45 AM) 75 mins

This exam is out of **100 points**

Name: _____

Net-ID: _____

1 Perceptron *(20 points)*

Consider the dataset $\mathcal{D} = ([x_1, x_2], y)$ containing all points $x_1 \in \{1, 2, 3\}$ and $x_2 \in \{1, 2, 3\}$. y is equal to 1 if the sum $x_1 + x_2$ is evenly divisible by 3 and 0 otherwise.

1. *(4 points)* Are the points linearly separable using features x_1 and x_2?

   Solution. No, the points are not linearly separable.

2. *(6 points)* Assume that the current decision boundary's normal weight vector is pointing in direction $(1, 4)$ and that the decision boundary passes through the point $(0, 0)$. How will the boundary move if the next point to be analyzed by the perceptron is $(x_1 = 3, x_2 = 3)$?

   Solution. The decision boundary will not move as this point is classified correctly.

3. *(4 points)* What happens if the point analyzed after that is $(x_1 = 1, x_2 = 1)$?

   Solution. The decision boundary will move according to the update function in the direction of its normal vector to accommodate for the incorrectly classified point.

4. *(6 points)* For these data, will the perceptron algorithm converge? Please provide your reason.

Solution. No, the perceptron algorithm will never converge for this set of points as they are not linearly separable and this is a requirement for convergence of perceptron.

2 Logistic Regression/Cross-Validation *(20 points)*

Given a training set $\mathcal{D} = \{(x_i, y_i | x_i \in \mathbf{R}^p, y_i \in \{0,1\}$, where $i \in \{1,..,n\}\}$. Here x_i's are p-dimensional feature vectors and y_i's are binary labels. We want to find the parameters $\hat{w}$ which maximize the likelihood for the training set, assuming a parametric model of the form: $p(y=1|x;w) = \frac{1}{(1+exp(-w^T x))}$. The conditional log likelihood of the training set is $l(w) = \sum_{i=1}^{n} y_i \; log \; p(y_i = 1|x_i; w) + (1-y_i) \; log \; (1 - p(y_i = 1|x_i; w))$. Thus

$$w^* \in argmax_w \sum_{i=1}^{n} (y_i) log[p(y_i = 1|x_i; w)] + (1-y_i) log[1 - p(y_i = 1|x_i; w)]$$

1. *(10 points).* Consider the case with binary features, i.e, $x \in \{0,1\}^p$ where feature x_1 happens to appear in the training set with only value 1. Is the norm of the gradient ever zero for any finite w?

   Solution. The gradient is

   $$\nabla_w L = \sum_{i=1}^{n} (y_i - \frac{1}{1 + exp(-\mathbf{w}^T \mathbf{x_i})}) \mathbf{x_i}$$

   Based on the gradient, we can construct a special case where the gradient is zero. Suppose $x_1 = x_2 = ... = x_p = 1$ for all $\mathbf{x_i}$ and $w_1 = w_2 = ... = w_p = w_0$. When we set gradient to zero, we can get

   $$\sum_{i=1}^{n} y_i = \frac{n}{1 + exp(-pw_0)}$$

   Accordingly, the can get

   $$w_0 = -\frac{log(n/(\sum_{i=1}^{n} y_i) - 1)}{p}$$

   w_0 is proper as long as $\sum_{i=1}^{n} y_i \neq n$. Therefore, the correct answer here is YES. However, since the condition for the gradient being zero is very strict, this can happen with probability 0. Also, constructing a counterexample is really hard in an exam. As a result, we also give you full credit if you claim the gradient can never be zero and prove it for the gradient of single data points. Specifically, the gradient for single data is

   $$\nabla_w L_i = (y_i - \frac{1}{1 + exp(-\mathbf{w}^T \mathbf{x_i})}) \mathbf{x_i}$$

   Therefore,

   $$||\nabla_w L_i||^2 = (y_i - \frac{1}{1 + exp(-\mathbf{w}^T \mathbf{x_i})})^2 ||\mathbf{x_i}||^2$$

   Because the norm of w is finite, $(y_i - \frac{1}{1+exp(-\mathbf{w}^T \mathbf{x_i})})^2 > 0$. Also, $||\mathbf{x_i}||^2 > 0$ as the first feature of $\mathbf{x_i}$ is 1. Consequently, $||\nabla_w L_i||^2 > 0$ and $\nabla_w L_i \neq \mathbf{0}$.

For both cases, the grading relies heavily on the calculation of gradient. As long as you get the correct gradient, you can get most of the points.

2. *(10 points).* Write a new objective function by adding a regularization term to the conditional negative log likelihood. (Remember you should multiply the regularization term with regularization constant C). What technique(s) have you learned in class that would allow you to choose an optimal C? Would the value of the gradient change from part (b)?

   Solution.

   The regularization term is $\frac{C}{2}\mathbf{w}^T\mathbf{w}$. We can use nested cross validation to choose an optimal C. The gradient will change by adding $C\mathbf{w}$.

3 Decision Trees *(10 points)*

The following table contains training examples that help predict whether a patient is likely to have a heart attack:

Patient ID	**Chest-Pain?**	**Male?**	**Smokes?**	**Exercises?**	**Heart-Attack?**
1	yes	yes	no	yes	yes
2	yes	yes	yes	no	yes
3	no	no	yes	no	yes
4	no	yes	no	yes	no
5	yes	no	yes	yes	yes
6	no	yes	yes	yes	no

(10 points). Use classification accuracy as the splitting criteria to construct a decision tree that predicts whether or not a patient is likely to have a heart attack.

Solution. First split on chest pain then split on exercises.

(Note: multiple solutions possible)

4 Variable Importance *(35 Points)*

In random forests, there is a way of measuring variable importance. The core of that method involves permuting the values of the jth feature. To do this, one would take the data matrix X and use a function that randomly permutes the values in column j. Answer the following about this idea:

1. *(4 points).* (True/False) Permuting feature j in data X would alter the association between feature j and outcome Y.

   Solution. True.

2. *(4 points).* (True/False) Permuting feature j in data X would alter the joint distribution of the features, $pdf(X_1, X_2, ..., X_p)$.

   Solution. True.

3. *(4 points).* (True/False) Permuting feature j in data X would alter the marginal distribution of the feature X_j, $pdf(X_j)$.

   Solution. False.

   Let us consider *model reliance* (also called permutation importance, and it also has several other names), which was defined in class. This measures how much the loss increases when we permute variable j. We can define it formally. Take the data matrix X, and permute its jth column, which corresponds to the jth feature. Define the permuted data matrix to be $\tilde{X}$. Row i of the new data matrix is $\tilde{x}_i$. Model reliance is:

   $$\frac{\sum_{i=1}^{n} \text{loss}(f(\tilde{x}_i), y_i)}{\sum_{i=1}^{n} \text{loss}(f(x_i), y_i)}.$$

4. *(8 points).* (True/False) Model reliance measures how important variable j is generally, in a model independent way.

   Solution. False. It depends on model f.

5. *(15 points).* Provide sufficient conditions on data $\{(x_i, y_i)\}_i$ and model f under which model reliance of feature j is approximately equal to the coefficient of feature j in linear regression. These just need to be sufficient conditions, not necessary conditions.

   Solution.

   - The data need to be normalized i.e. zero mean and unit variance to avoid bias due to units/scale of the covariates.
   - None of the covariates have low-variance problem. In other words, probability of realizing almost all possible values of a certain covariate j in the observed dataset is significant and no value dominates the occurences.
   - All the covariates are uncorrelated with each other.
   - (optional) y_i's are determined by linear combination of corresponding features in x_i.
   - (optional) $x_i \overset{\text{iid}}{\sim} \mathcal{N}(0_p, \mathbf{I}_{p\times p})$

5 Boosting *(15 points)*

1. *(5 points).* (True/False) AdaBoost will eventually give zero training error regardless of the type of weak classifier it uses, provided enough iterations are performed.

   Solution. False. Not if the data in the training set cannot be separated by a linearcombination of the specific type of weak classifiers we are using. For example consider the XOR example with decision stumps as weak classifiers. No matter how many iterations are performed zero training error will not be achieved.

2. *(5 points).* (True/False) Within an iteration of AdaBoost, the weights of all the misclassified examples go up by the same multiplicative factor.

   Solution. True. The multiplying factor is equal to $\frac{1}{\epsilon_t}$ or $\frac{e^{\alpha_t}}{Z_t}$ for the model we have discussed in class and discussion sections. This is same for all misclassified examples.

3. *(5 points).* (True/False) In AdaBoost, weighted training error ϵ_t of the t-th weak classifier on training data with weights $d_{t,i}$ $\forall i \in \{1..n\}$ tends to decrease as a function of t.

   Solution. False. In the course of boosting iterations the weak classifiers are forced to try to classify more difficult examples. The weights will increase for examples that are repeatedly misclassified by the weak classifiers. The weighted training error ϵ_t of the t-th weak classifier on the training data therefore might increase sometimes.