Solutions

10-701 Machine Learning **Name:** ______________________
Fall 2019
Midterm Exam
10/21/2019
Time Limit: 120 minutes **Andrew ID** ______________________

Instructions:

- Fill in your name and Andrew ID above. Be sure to write neatly, or you may not receive credit for your exam.
- This exam contains 21 pages (including this cover page and 2 blank pages at the end). There are 8 sections.
  The total number of points is 100.
- Clearly mark your answers in the allocated space. If your solution is messy please highlight the answer so it is clearly visible. If you have made a mistake, cross out the invalid parts of your solution, and circle the ones which should be graded. If given a choice please fill in the circle next to the answer you want to select.
- Look over the exam first to make sure that none of the 21 pages are missing. The problems are of varying difficulty, so you may wish to pick off the easy ones first.
- You may use a scientific calculator for this exam, however no other electronic devices can be used.
- Please write all answers in pen.
- You have 120 minutes to complete the exam. Good luck!

Topic	**Section**	**Points**
Probability	1	16
Density estimation and MLE	2	10
Logistic Regression vs. Gaussian Naive Bayes	3	10
Decision Trees	4	16
Perceptron	5	12
Neural Networks	6	14
SVMs	7	10
Adaboost	8	12
—	**Total**	100

1 Probability (16 Points)

1. Suppose $X_1, X_2, ..., X_n$ are i.i.d. samples from a Gaussian distribution $\mathcal{N}(\mu^*, 1)$, where $\mu^* = -1$. Mary has access to this sample set and she is aware that $\{X_i\}_{i=1}^n$ comes from a Gaussian distribution with variance 1, however Mary does not know the true mean of this dataset. To estimate the true mean of the distribution that the dataset comes from Mary assumes that these samples are from $\mathcal{N}(\hat{\mu}, 1)$, where $\hat{\mu}$ is a random variable.

   (a) [4 pts] **Choose one:** If she assumes $\hat{\mu} \sim \mathcal{N}(0, 1)$, as the sample size increases, the posterior density at $\hat{\mu}$ will:

   ◯ Concentrate around -1 with the density value at -1 not exceeding 1.

   ◯ Concentrate around -1 with the density value at -1 exceeding 1.

   ◯ Concentrate around 0 with the density value at one exceeding 1.

   ◯ Concentrate around 0 with the density value at one not exceeding 1.

   In one or two short sentences explain your choice:

   The answer is (b). Notice that as they data increases since the prior has a nonzero mass on $\hat{\mu} = -1$, the posterior density will converge to a density to concentrates its mass over -1, which is the true value of μ^*.

   (b) [4 pts] **Choose one:** If Mary assumes $\hat{\mu} \sim \mathcal{U}(0, 1)$ where the latter is uniform distribution on interval $[0, 1]$ then the posterior distribution on $\hat{\mu}$ will: *Hint: Using Bayes' theorem see where the posterior density on $\hat{\mu}$ attains the value zero.*

   ◯ Concentrate around -1 with the density value at -1 not exceeding 1.

   ◯ Concentrate around -1 with the density value at -1 exceeding 1.

   ◯ Remains the uniform distribution $\mathcal{U}(0, 1)$.

   ◯ None of the above

   In one or two short sentences explain your choice:

   The answer is (d). Note that this time the prior mass on $\hat{\mu} = 0$ is zero. As such using Bayes' theorem we have:

$$\forall \mu \in (-\infty, 0) \quad p(\hat{\mu} = \mu | \mathcal{D}) = \frac{p(\mathcal{D}|\hat{\mu} = \mu) p(\hat{\mu} = \mu)}{p(\mathcal{D})} = 0,$$

since $p(\hat{\mu} = \mu) = 0$. So the posterior on $\hat{\mu} = 0$ and anywhere close to $\hat{\mu} = 0$ (in fact anywhere where $\hat{\mu} \in (-\infty, 0)$ is zero, with the same reasoning above). (c) is not the answer either as for any μ_1 and μ_2

$$\frac{p(\hat{\mu} = \mu_1 | \mathcal{D})}{p(\hat{\mu} = \mu_2 | \mathcal{D})} = \frac{p(\mathcal{D} | \hat{\mu} = \mu_1)}{p(\mathcal{D} | \hat{\mu} = \mu_2)},$$

which is obviously not constant as opposed to what the choice (c) implies. Therefore (d) is the solution.

Naji

2. For a given n suppose we build the vectors $\mathbf{w}_n$ and $\mathbf{v}_n$ s.t.

$$\mathbf{w}_n = (X_1, ..., X_n) \text{ and } \mathbf{v}_n = (Y_1, ..., Y_n)$$

where $X_i \sim \mathcal{N}(0, \frac{1}{n})$ and $Y_i \sim \mathcal{N}(0, \frac{1}{n})$ for all i where X_i's and Y_j's are mutually independent, i.e. every X_i is independent from every X_j ($j \neq i$) and every Y_k. Also every Y_i is independent from every Y_j ($i \neq j$).

(a) **[4 pts]** Derive the expectation of the inner product of $\mathbf{w}_n$ and $\mathbf{v}_n$, i.e. $\mathbf{w}_n \cdot \mathbf{v}_n$.
We have the following

$$\mathbb{E}(\mathbf{W}_n \cdot \mathbf{V}_n) = \mathbb{E}(\sum_{i=1}^{n} X_i Y_i) \overset{*}{=} \sum_{i=1}^{n} \mathbb{E}(X_i Y_i) \overset{**}{=} \sum_{i=1}^{n} \mathbb{E}(X_i)\mathbb{E}(Y_i) = 0$$

Notice that this is independant of n and is true for any n.

(b) **[4 pts]** Derive the expectation of the norm-squared of $\mathbf{v}_n$ and $\mathbf{w}_n$, i.e. $\|\mathbf{w}_n\|_2^2$ and $\|\mathbf{v}_n\|_2^2$.

$\mathbf{w}_n$ and $\mathbf{v}_n$ are similar so we calculate the length for $\mathbf{w}_n$ and for $\mathbf{v}_n$ it will be the same:

$$\mathbb{E}(\|\mathbf{W}_n\|^2) = \mathbb{E}(\sum_{i=1}^{n} X_i^2) = \sum_{i=1}^{n} \mathbb{E}(X_i^2) = \sum_{i=1}^{n} \mathbb{V}\text{ar}(X_i) = n * 1/n = 1$$

Naji

2 Density estimation and MLE (10 Points)

We first define the Pareto distribution as:

$$p(x|k,\alpha) = \begin{cases} \frac{\alpha k^\alpha}{x^{\alpha+1}} & x \in [k, \infty) \\ 0 & \text{otherwise} \end{cases}$$

$$k, \alpha \in (0, \infty)$$

1. **[5 pts]** You have n independent samples $x_1, x_2, ..., x_n$ drawn from a Pareto distribution. What is the MLE for the parameters k and α?

$$\hat{k}_{MLE} = \underline{\hspace{4cm}}$$

$$\hat{\alpha}_{MLE} = \underline{\hspace{4cm}}$$

$$\hat{k}_{MLE} = \min_i x_i$$

$$\hat{\alpha}_{MLE} = \frac{n}{\sum_{i=1}^n \ln\left(\frac{x_i}{\hat{k}_{MLE}}\right)}$$

2. **[5 pts]** We re-parametrize the Pareto distribution such that $\alpha = e^p$. Given that $k = \sqrt{2}$, what is the MLE for p?

$$\hat{p}_{MLE} = \ln\left(\frac{n}{\sum_{i=1}^n \ln\left(\frac{x_i}{\sqrt{2}}\right)}\right)$$

3 Logistic Regression vs. Gaussian Naive Bayes (10 Points)

For a given binary classifier, where Y is the binary label and $\mathbf{X}$ represents the feature vector $\mathbf{X} = (X_1, ..., X_n)$ with conditional likelihood $P(Y|\mathbf{X})$, the decision function is defined as

$$\delta(\mathbf{x}^*) = \frac{p(Y = 1|\mathbf{X} = \mathbf{x}^*)}{p(Y = 0|\mathbf{X} = \mathbf{x}^*)}.$$

Notice that for a given $\mathbf{x}^*$, our classifier returns the label 1 if $\delta(\mathbf{x}^*) \geq 1$ and zero otherwise. One calls a classifier linear when whether $\delta(\mathbf{x}) \geq 1$ or not is a linear function of the features $\mathbf{X}$.

1. [**5 pts**] Show Logistic regression is a linear classifier.
   We have

$$\log(\delta(\mathbf{x}^*)) = \log\left(\frac{p(Y = 1|\mathbf{X} = \mathbf{x}^*)}{p(Y = 0|\mathbf{X} = \mathbf{x}^*)}\right) = -\log\left(\exp(w^T\mathbf{X}^* + b)\right) = -w^T\mathbf{X}^* - b$$

   which means $\log(\delta(\mathbf{x}^*)) \geq 0$ if and only if $w^T\mathbf{X}^* + b \leq 0$.

2. [**5 pts**] Show that Gaussian Naive Bayes has a linear decision surface when the class conditional distributions ($P(\mathbf{X}|Y = i)$ $i = 0, 1$) have equal covariance matrix. You can assume class priors are the same.

   We have

$$\log(\delta(\mathbf{x}^*)) = \log\left(\frac{p(Y = 1|\mathbf{X} = \mathbf{x}^*)}{p(Y = 0|\mathbf{X} = \mathbf{x}^*)}\right) = \log\left(\frac{p(\mathbf{X} = \mathbf{x}^*|Y = 1)}{p(\mathbf{X} = \mathbf{x}^*|Y = 0)}\right) =$$
$$(\mathbf{x}^* - \mu_0)'\Sigma^{-1}(\mathbf{x}^* - \mu_0) - (\mathbf{x}^* - \mu_1)'\Sigma^{-1}(\mathbf{x}^* - \mu_1) =$$
$$-\mathbf{x}^{*T}\Sigma^{-1}\mathbf{x}^* + 2\mathbf{x}^{*T}\Sigma^{-1}\mu_0 - \mu_0^T\Sigma^{-1}\mu_0 + \mathbf{x}^{*T}\Sigma^{-1}\mathbf{x}^* - 2\mathbf{x}^*\Sigma^{-1}\mu_1 + \mu_1{}^T\Sigma^{-1}\mu_1,$$

   which is a linear function of $\mathbf{x}^*$.

Naji

4 Decision Trees (16 Points)

1. We would like to learn a decision tree. We have n samples for training. You can assume that n is large and we are using continuous features. In the following questions we would only use the first feature, x_1, in each of the samples for splitting them.

   (a) [**3 pts**] **Choose one:** We would like to split according to x_1 at the root with 3 branches (samples are split at the root to three different sub-trees which by findings values a and b such that the three sub-trees are $x_1 < a$, $a <= x_1 <= b$ and $x_1 > b$). What is the runtime for finding the values of x_1 that should be used by the root for such split?

   ◯ $O(\sqrt{n})$

   ◯ $O(n)$

   ◯ $O(n^2)$

   ◯ $O(n^3)$

   ◯ Infinite

   C. To split on continues values we need to find the best thresholds a and b. This requires as to order the values of x_1 and then test all possible splits and so the runtime is $O(n^2)$.

   (b) [**4 pts**] **Choose one:** Following our first split, we would like to *split again* on x_1 on each of the three sub-trees resulting from the split in the previous question. Again, for each sub-tree we would split three ways as we did in the root. What is the total run-time for determining the optimal splits for ALL three sub-trees?

   ◯ $O(n)$

   ◯ $O(n^2)$

   ◯ $O(n^3)$

   ◯ $O(n^4)$

   ◯ $O(n^8)$

   ◯ Infinite

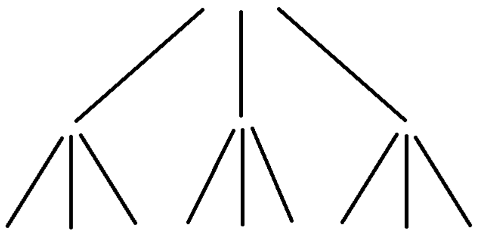
   ◯ Impossible to tell

   B. Let the number of samples assigned to each of the three sub-trees be n_1, n_2 and n_3. We know from question 1 that the total run time for each is $O(n_1^2)$, $O(n_2^2)$, $O(n_3^2)$. Since $n_1 + n_2 + n_3 = n$ one of these sets is $O(n)$ and so the total runtime is $O(n^2)$.

   (c) [**4 pts**] For the same data we would like to learn the *best* tree that:

   - Only uses x_1

- Has two levels, a root and a level below it where each splits to three branches (see figure on the next page).

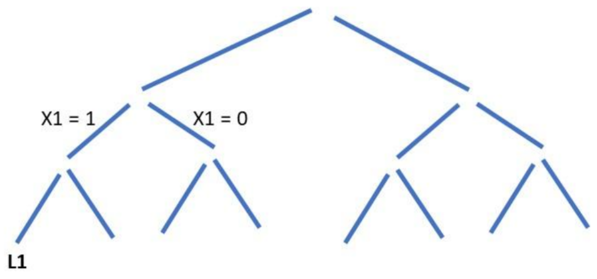


Choose one: What is the runtime for computing the *optimal tree* for such parameters?

- ◯ $O(n)$
- ◯ $O(n^2)$
- ◯ $O(n^4)$
- ◯ $O(n^8)$
- ◯ Infinite
- ◯ Impossible to tell

D. Finding the best way to split in this case is similar to finding the best 9 way split. This means that the total run time is $O(n^8)$.

2. Assume we are learning a decision tree for two classes from a dataset with a large number of binary features. We decide to learn a tree of depth at most 3 which means that in many of the leaves there are samples from both classes. To classify at the leaves, we learn a Nave Bayes classifier for each leaf and every sample that reaches a leaf is classified using the classifier learned for that leaf.



As can be seen in the figure, we have split on X_1 on the way to the leaf L1.

(a) [**5 pts**] Does this create a problem for the Nave Bayes classifier we learn for L1 in this case? If so, what is the problem? If not, why not? No. While

$P(X_1 = 0|C = 1)$ is 0 for L1, thats not a problem since only samples with $X_1 = 1$ would be assigned to that node. So the probability for either of the classes would not be set to 0 because of the lack of 0 values for X_1.

5 Perceptron (12 Points)

1. [**4 pts**] Consider data points $\{x^{(i)}\}_{i=1}^n$ and their labels $\{y^{(i)}\}_{i=1}^n$, where $x^{(i)} \in \{0,1\}^d$ $(d \geq 2), y^{(i)} \in \{0,1\},\ \forall i \in \{1,\ldots,n\}$.

   Suppose the dataset is generated using AND function, which means

   $$y^{(i)} = x_1^{(i)} \wedge x_2^{(i)} \wedge \ldots \wedge x_d^{(i)}.$$

   Is it possible to use perception algorithm to classify this dataset? Explain your answer.
   It is possible. The only data point with $y = 1$ is when $x_1 = x_2 = \ldots = x_d = 1$, and all

   other data points belong to another class. Therefore the data is linear separable. An example hyper-plane to separate the data is $x_1 + x_2 + \ldots + x_d = d - 0.5$.

   The perception algorithm is guaranteed to terminate in this case, with all data points end up being classified.

2. [**8 pts**] Now consider another set of points $\{(x_1^{(i)}, x_2^{(i)}, y^{(i)})\}_{i=1}^n$. In this setting, $(x_1^{(i)}, x_2^{(i)})$ are drawn from one of the two quadratic functions depicted below, with the constraint that $x_1 \neq 0$. In other words, each point is labeled as either red or blue, i.e., $y^{(i)} \in \{red, blue\}$, and blue points are sampled from the blue quadratic function while red points are sampled from the red quadratic function (again, with the constraint that $x_1 \neq 0$).

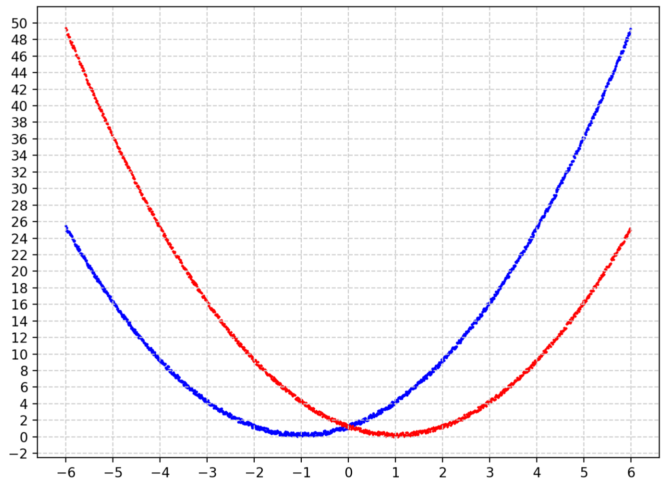


Figure 1: A plot of the data, with x_1 on the horizontal axis and x_2 on the vertical axis

Is it possible to run perception algorithm to classify it? If so, explain why. If not, provide a function $z = f(x_1, x_2)$ that transforms $\{(x_1^{(i)}, x_2^{(i)})\}_{i=1}^n$ to $\{z^{(i)}\}_{i=1}^n$ such that it is possible to run the perceptron on z.

No. Clearly the data is not separable.

Notice that the red and blue data approximately follow the equations:

$$red: \ x_2 = (x_1 - 1)^2, \quad blue: \ x_2 = (x_1 + 1)^2.$$

Therefore we can have the transformation

$$z = f(x_1, x_2) = \frac{x_2 - x_1^2 - 1}{x_1}.$$

After transformation, all data points with blue label have values z approximately 2, and all data points with red label have values z approximately -2. Therefore now it is linearly separable and we can apply perception algorithm to classify them. Note that we need to assume that $x_1 \neq 0$ b/c otherwise the point (0,1) is ambiguous to classify.

6 Neural Networks (14 Points)

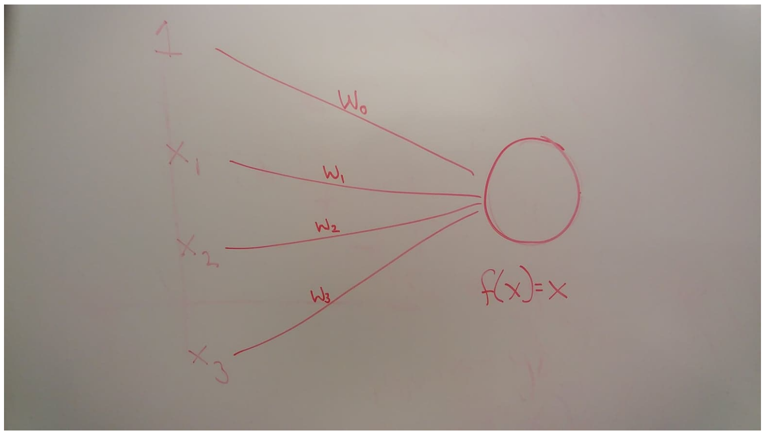
1. Imagine we have a dataset X which has 10 datapoints each with 3 features (x_1^i, x_2^i, x_3^i) for $i \in \{1, 2, 3, ..., 10\}$ and corresponding labels $y = (y^1, y^2, ...y^{10})$.

   (a) **[4 pts]** In the space below draw the smallest neural network which can fully represent the Linear Regression model learned in class on X, your drawing should be fully labelled. In the boxes below you should define what the Activation/transfer functions and loss function should be in your model.

   Activation/Transfer function:

   Loss function:

   The students solution should look something like this and be fully labelled as shown. It is important that the student considers the bias term.



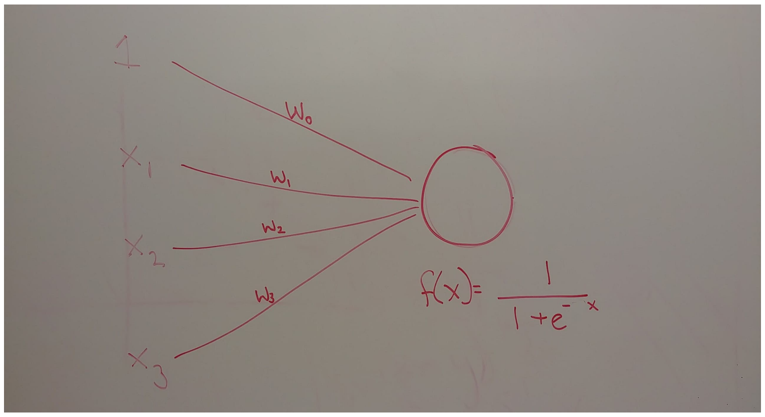
   With Transfer function $f(x) = x$ and Loss function MSE

(b) [**4 pts**] In the space below draw the smallest neural network which can fully represent the Logistic Regression model learned in class on X, your drawing should be fully labelled. In the boxes below you should define what the activation/transfer functions and loss function should be in your model.

Activation/Transfer function:

Loss function:

The students solution should look something like this and be fully labelled as shown. It is important that the student considers the bias term.



With Transfer function being the Sigmoid function and Cross Entropy Loss.

2. [**6 pts**] Your friend runs a Neural Network on training data and gains an almost perfect training accuracy however they perform very poorly when it comes to test time, you conclude that their model is overfitting. In the below boxes choose from **{Increase, Decrease or N/A}** to indicate what they should do with their model to reduce the amount of overfitting that your friend is causing. (N/A here means changing it won't

effect overfitting) *You should assume the training and test data are representative of the true underlying distribution of the data.*

	the number of training data.
	the number of test data.
	the regularization constant.
	the number of hidden layers
	the number of neurons
	the number of outputs

Increase
N/A
Increase
Decrease
Decrease
N/A

7 SVMs (10 Points)

Consider the following primal and dual forms of SVM.

Primal	Dual
$\min_{\boldsymbol{w},b} \left\{ \frac{1}{2} \lVert \boldsymbol{w} \rVert_2^2 + C \sum_{i=1}^n \xi_i \right\}$	$\max_{\boldsymbol{\alpha}} \left\{ \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j} y_i y_j \alpha_i \alpha_j \mathbf{x}_i^\top \mathbf{x}_j \right\}$
s.t. $\forall i,\ y_i(\boldsymbol{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0.$	s.t. $\forall i,\ 0 \leq \alpha_i \leq C, \quad \sum_{i=1}^n \alpha_i y_i = 0.$

where $\boldsymbol{w}, \mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{\pm 1\}$ and n denotes the number of data points. The dual variables are represented by α_i and $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j$. Provide short answers to the following questions about SVM:

1. [**4 pts**] Suppose you wish to use SVMs to solve a learning problem where some training data points are more important than others. More formally, assume that each training point consists of a triplet (x_i, y_i, p_i), where $0 \leq p_i \leq 1$ is the importance of the ith point. Rewrite the primal soft-margin SVM constrained optimization problem so that the penalty for mis-labeling a point x_i is scaled by the priority p_i. The modified primal

   optimization problem can be written as

$$\begin{aligned} \text{minimize} \quad & \tfrac{1}{2}||w||^2 + C \textstyle\sum_{i=1}^n \xi_i p_i \\ \text{subject to} \quad & y_i[w \cdot x_i + b] \geq 1 - \xi_i, \quad \xi_i \geq 0\,. \end{aligned}$$

2. [**6 pts**] Write down the Lagrangian for this problem and compute partial derivatives $\frac{\partial L}{\partial w}$, $\frac{\partial L}{\partial b}$, and $\frac{\partial L}{\partial \xi_i}$. How (if at all) do they differ from the partial derivatives for standard non-separable SVM optimization problem?

The Lagrangian holding for all $w, b, \alpha_i \geq 0, \beta_i \geq 0$ is then

$$\begin{aligned} L(w, b, \alpha) &= \frac{1}{2}||w||^2 + C\sum_{i=1}^{m} \xi_i p_i \\ &\quad - \sum_{i=1}^{m} \alpha_i [y_i(w \cdot x_i + b) - 1 + \xi_i] - \sum_{i=1}^{m} \beta_i \xi_i \,. \end{aligned} \tag{1}$$

$\frac{\partial L}{\partial w}$ and $\frac{\partial L}{\partial b}$ are the same as for the regular non-separable SVM optimization problem, while $\frac{\partial L}{\partial \xi_i} = Cp_i - \alpha_i - \beta_i$, which differs by the factor p_i.

8 Adaboost (12 Points)

Following the notation from class, let h_t be the weak learner selected by Adaboost at round t, and define the contribution of the t-th weak learner to the ensemble model as β_t, and the weighted classification error of h_t as ϵ_t.

1. **[3 pts]** Show that $(1 - \epsilon_t)e^{-\beta_t} = \epsilon_t e^{\beta_t}$.

   The result follows directly from the given equation: $\beta_t = \frac{1}{2}\log\frac{1-\epsilon_t}{\epsilon_t}$. Simply exponentiate both sides and rearrange terms.

2. **[3 pts]** Using the result from part (a), show that

$$\sum_{n, y_n \neq h_t(x_n)} w_{t+1}(n) = \sum_{n, y_n = h_t(x_n)} w_{t+1}(n)$$

   .

   Observe that $\sum_{n, y_n \neq h_t(x_n)} w_{t+1}(n) = \epsilon_t e^{\beta_t}$ and $\sum_{n, y_n = h_t(x_n)} w_{t+1}(n) = (1 - \epsilon_t)e^{-\beta_t}$. Thus, from part (a) we have equality.

3. **[3 pts]** What is the significance of the result you proved in the previous question?

   This result shows that Adaboost puts equal total weight at round $t + 1$ on correctly and incorrectly classified training instances at round t.

4. **[3 pts]** Imagine after t rounds that the performance of Adaboost on the training data is perfect, i.e., we have zero training error. Does this imply that $\epsilon_{t+1} = 0$? Why / Why

not?

Training error at time t is a property of the ensemble classifier on the uniform distribution over the training data, whereas ϵ_{t+1} is a property of an individual weak learner on the distribution of the training data after round t. Hence one does NOT imply the other.

Do not remove this page! Use this page for scratch work.

Do not remove this page! Use this page for scratch work.