Solutions

10-601 Machine Learning | **Name:**
Spring 2020 | **Andrew Email:**
Exam 1 Practice Problems | **Room:**
March 27, 2020 | **Seat:**
Time Limit: N/A | **Exam Number:**

Instructions:

- Fill in your name and Andrew ID above. Be sure to write neatly, or you may not receive credit for your exam.
- Clearly mark your answers in the allocated space **on the front of each page.** If needed, use the back of a page for scratch space, but you will not get credit for anything written on the back of a page. If you have made a mistake, cross out the invalid parts of your solution, and circle the ones which should be graded.
- No electronic devices may be used during the exam.
- Please write all answers in pen.
- You have N/A to complete the exam. Good luck!

Topic	**Pages**	**# Questions**
Logistic Regression	2	3
Regularization	4	1
Neural Network	5	3
PAC Learning	10	2
MLE/MAP	12	8
Naive Bayes	15	7

1 Logistic Regression

1. **[2 pts]** If today I want to predict the probability that a student sleep more than 8 hours on average (SA) given the Course loading (C), I will choose to use linear regression over logistic regression.

   Circle one: True False

   False.

2. Answer the following questions with brief explanations where necessary.

   a) **[2 pts]** A generalization of logistic regression to a multiclass settings involves expressing the per-class probabilities $P(y = c|x)$ as the softmax function $\frac{\exp(w_c^T x)}{\sum_{d \in C} \exp(w_d^T x)}$, where c is some class from the set of all classes C.

   Consider a 2-class problem (labels 0 or 1). Rewrite the above expression for this situation, to end up with expressions for $P(Y = 1|x)$ and $P(Y = 0|x)$ that we have already come across in class for binary logistic regression.

   $P(y = 1|x) = \frac{\exp(w_1^T x)}{\exp(w_0^T x)+\exp(w_1^T x)} = \frac{\exp((w_1-w_0)^T x)}{1+\exp((w_1-w_0)^T x)} = \frac{\exp(w^T x)}{1+\exp(w^T x)} = p$

   Therefore, $1 - p = \frac{1}{1+\exp(w^T x)}$

   b) **[3 pts]** Given 3 data points $(1, 1), (1, 0), (0, 0)$ with labels $0, 1, 0$ respectively. Consider 2 models, Model 1: $\sigma(w_1x_1 + w_2x_2)$, Model 2: $\sigma(w_0 + w_1x_1 + w_2x_2)$ ($\sigma(z)$ is the sigmoid function $\frac{1}{1+e^{-z}}$) that compute $p(y = 1|\mathbf{x})$. Using the given data, we can learn parameters $\hat{w}$ by maximizing the conditional log-likelihood.

   Suppose we switched $(0, 0)$ to label 1 instead.

   Do the parameters learnt for Model 1 change?

   Circle one: **True** **False**
   One-line explanation:

   False. The parameters learnt for Model 1 don't change because $w_1x_1 + w_2x_2 = 0$ for (0, 0). Hence $p = 0.5$ irrespective of the labels or the values of w.

   What about Model 2?

   Circle one: **True** **False**
   One-line explanation:

   True. This model has a bias term which remains non-zero for (0, 0), and can thus change the model depending on the label assigned.

   c) **[2 pts]** For logistic regression, we need to resort to iterative methods such as gradient descent to compute the $\hat{w}$ that maximizes the conditional log likelihood. Why?

   There is no closed-form solution.

d) [**3 pts**] Considering a Gaussian prior, write out the MAP objective function $J(w)_{MAP}$ in terms of the MLE objective $J(w)_{MLE}$. Name the variant of logistic regression this results in.

$J_{MAP}(\mathbf{w}) = J_{MLE}(\mathbf{w}) - \lambda\|\mathbf{w}\|_2^2$. This is L2 regularized logistic regression.

3. Given a training set $\{(x_i, y_i), i = 1, \dots, n\}$ where $x_i \in \mathbb{R}^d$ is a feature vector and $y_i \in \{0, 1\}$ is a binary label, we want to find the parameters $\hat{w}$ that 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

$$\ell(w) = \sum_{i=1}^{n} y_i \log p(y_i, |x_i; w) + (1 - y_i) \log(1 - p(y_i, |x_i; w)),$$

and the gradient is

$$\nabla\ell(w) = \sum_{i=1}^{n} (y_i - p(y_i|x_i; w))x_i.$$

a) [5 pts.] Is it possible to get a closed form for the parameters $\hat{w}$ that maximize the conditional log likelihood? How would you compute $\hat{w}$ in practice?

There is no closed form expression for maximizing the conditional log likelihood. One has to consider iterative optimization methods, such as gradient descent, to compute $\hat{w}$.

b) [5 pts.] For a binary logistic regression model, we predict $y = 1$, when $p(y = 1|x) \geq 0.5$. Show that this is a linear classifier.

Using the parametric form for $p(y = 1|x)$:

$$\begin{aligned} p(y = 1|x) \geq \frac{1}{2} &\implies \frac{1}{1 + \exp(-w^T x)} \geq \frac{1}{2} \\ &\implies 1 + \exp(-w^T x) \leq 2 \\ &\implies \exp(-w^T x) \leq 1 \\ &\implies -w^T x \leq 0 \\ &\implies w^T x \geq 0, \end{aligned}$$

so we predict $\hat{y} = 1$ if $w^T x \geq 0$.

c) Consider the case with binary features, i.e, $x \in \{0, 1\}^d \subset \mathbb{R}^d$, where feature x_1 is rare and happens to appear in the training set with only label 1. What is $\hat{w}_1$? Is the gradient ever zero for any finite w? Why is it important to include a regularization term to control the norm of $\hat{w}$?

If a binary feature fired for only label 1 in the training set then, by maximizing the conditional log likelihood, we will make the weight associated to that feature be infinite. This is because, when this feature is observed in the training set, we will want to predict predict 1 irrespective of everything else. This is an undesired behaviour from the point of view of generalization performance, as most likely we do not believe this rare feature to have that much information about class 1. Most likely, it is spurious co-occurrence. Controlling the norm of the weight vector will prevent these pathological cases.

2 Feature Engineering and Regularization

1. **Model Complexity:** In this question we will consider the effect of increasing the model complexity, while keeping the size of the training set fixed. To be concrete, consider a classification task on the real line $\mathbb{R}$ with distribution D and target function $c^* : \mathbb{R} \to \{\pm 1\}$ and suppose we have a random sample S of size n drawn iid from D. For each degree d, let ϕ_d be the feature map given by $\phi_d(x) = (1, x, x^2, \ldots, x^d)$ that maps points on the real line to $(d+1)$-dimensional space.

   Now consider the learning algorithm that first applies the feature map ϕ_d to all the training examples and then runs logistic regression as in the previous question. A new example is classified by first applying the feature map ϕ_d and then using the learned classifier.

   a) [4 pts.] For a given dataset S, is it possible for the training error to increase when we increase the degree d of the feature map? **Please explain your answer in 1 to 2 sentences.** No. Every linear separator using the feature map ϕ_d can also be expressed using the feature map ϕ_{d+1}, since we are only adding new features. It follows that the training error must decrease for any given sample S.

   b) [4 pts.] Briefly **explain in 1 to 2 sentences** why the true error first drops and then increases as we increase the degree d. When the dimension d is small, the true error is high because it is not possible to the target function is not well approximated by any linear separator in the ϕ_d feature space. As we increase d, our ability to approximate c^* improves, so the true error drops. But, as we continue to increase d, we begin to overfit the data and the true error increases again.

3 Neural Networks

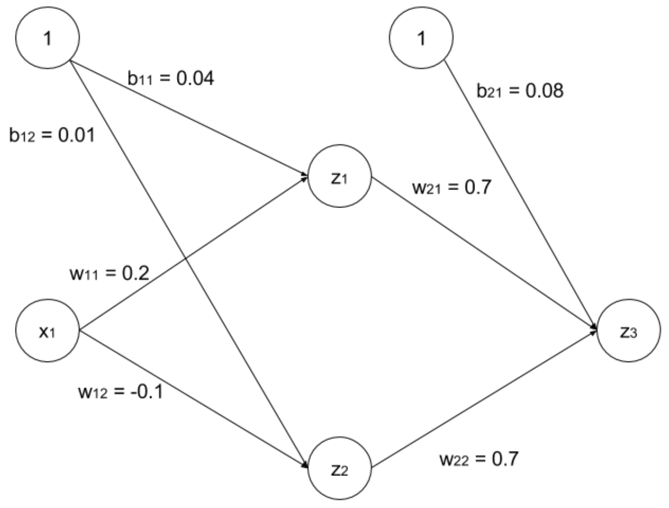


Figure 1: neural network

1. Consider the neural network architecture shown above for a 2-class $(0, 1)$ classification problem. The values for weights and biases are shown in the figure. We define:

$$a_1 = w_{11}x_1 + b_{11}$$
$$a_2 = w_{12}x_1 + b_{12}$$
$$a_3 = w_{21}z_1 + w_{22}z_2 + b_{21}$$
$$z_1 = \text{relu}(a_1)$$
$$z_2 = \text{relu}(a_2)$$
$$z_3 = \sigma(a_3),\ \sigma(x) = \frac{1}{1+e^{-x}}$$

Use this information to answer the questions that follow.

(i) **[6 pts]** For $x_1 = 0.3$, compute z_3, in terms of e. **Show all work.**
$z_3 =$

$z_3 = \frac{1}{1+e^{-0.15}}$

(ii) **[2 pts]** To which class does the network predict the given data point $(x_1 = 0.3)$, i.e., $\hat{y} =$? Note that $\hat{y} = 1$ if $z_3 > \frac{1}{2}$, else $\hat{y} = 0$.

Circle one: 0 1

$\hat{y}(x_1 = 0.3) = 1$

(iii) **[6 pts]** Perform backpropagation on the bias b_{21} by deriving the expression for the gradient of the loss function $L(y, z_3)$ with respect to the bias term b_{21}, $\frac{\partial L}{\partial b_{21}}$, in terms of the partial derivatives $\frac{\partial \alpha}{\partial \beta}$, where α and β can be any of $L, z_i, a_i, b_{ij}, w_{ij}, x_1$ for all valid values of i, j. Your backpropagation algorithm should be as explicit as possible—that is, make sure each partial derivative $\frac{\partial \alpha}{\partial \beta}$ cannot be decomposed further into simpler partial derivatives. Do *not* evaluate the partial derivatives.

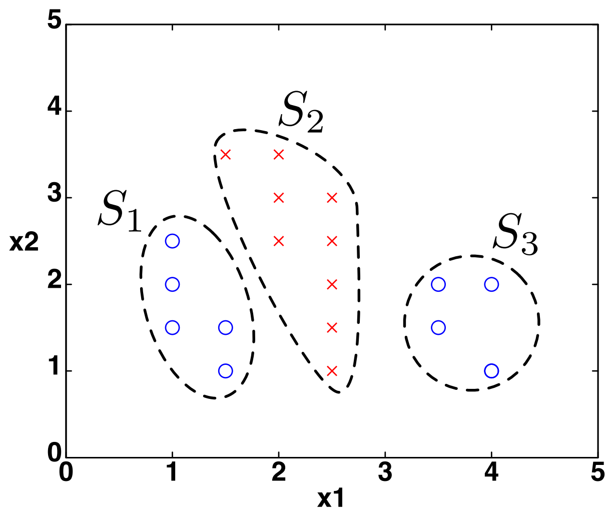
$$\frac{\partial L}{\partial b_{21}} = \frac{\partial L}{\partial z_3}\frac{\partial z_3}{\partial a_3}\frac{\partial a_3}{\partial b_{21}}$$

(iv) **[6 pts]** Perform backpropagation on the bias b_{12} by deriving the expression for the gradient of the loss function $L(y, z_3)$ with respect to the bias term b_{12}, $\frac{\partial L}{\partial b_{12}}$, in terms of the partial derivatives $\frac{\partial \alpha}{\partial \beta}$, where α and β can be any of $L, z_i, a_i, b_{ij}, w_{ij}, x_1$ for all valid values of i, j. Your backpropagation algorithm should be as explicit as possible—that is, make sure each partial derivative $\frac{\partial \alpha}{\partial \beta}$ cannot be decomposed further into simpler partial derivatives. Do *not* evaluate the partial derivatives.

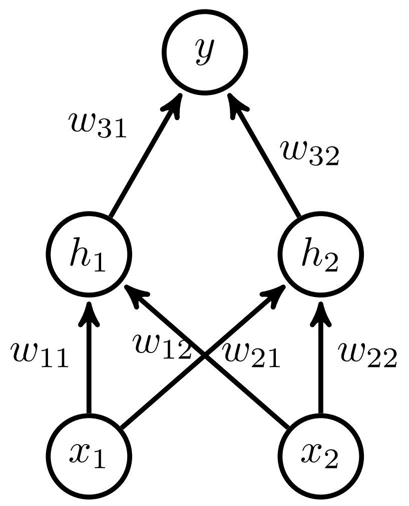
$$\frac{\partial L}{\partial b_{12}} = \frac{\partial L}{\partial z_3}\frac{\partial z_3}{\partial a_3}\frac{\partial a_3}{\partial z_2}\frac{\partial z_2}{\partial a_2}\frac{\partial a_2}{\partial b_{12}}$$

2. In this problem we will use a neural network to classify the crosses ($\times$) from the circles ($\circ$) in the simple dataset shown in Figure 2a. Even though the crosses and circles are not linearly separable, we can break the examples into three groups, S_1, S_2, and S_3 (shown in Figure 2a) so that S_1 is linearly separable from S_2 and S_2 is linearly separable from S_3. We will exploit this fact to design weights for the neural network shown in Figure 2b in order to correctly classify this training set. For all nodes, we will use the threshold activation function

$$\phi(z) = \begin{cases} 1 & z > 0 \\ 0 & z \le 0. \end{cases}$$

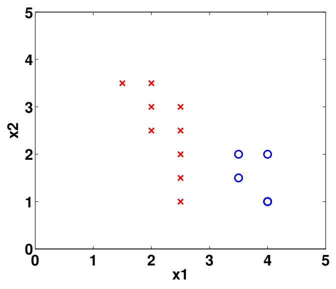


(a) The dataset with groups S_1, S_2, and S_3.

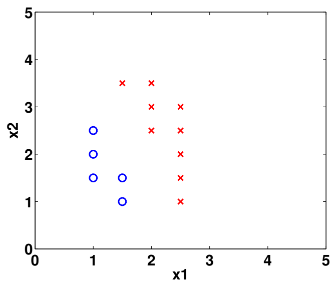


(b) The neural network architecture

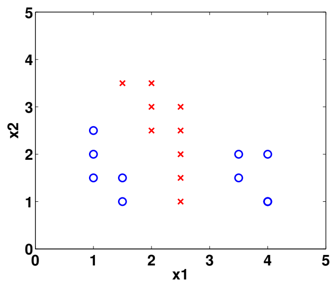
Figure 2



(a) Set S2 and S3



(b) Set S1 and S2



(c) Set S1, S2 and S3

Figure 3: NN classification.

(i) First we will set the parameters w_{11}, w_{12} and b_1 of the neuron labeled h_1 so that its output $h_1(x) = \phi(w_{11}x_1 + w_{12}x_2 + b_1)$ forms a linear separator between the sets S_2 and S_3.

(a) [1 pt.] On Fig 3a, draw a linear decision boundary that separates S_2 and S_3.

(b) [1 pt.] Write down the corresponding weights w_{11}, w_{12}, and b_1 so that $h_1(x) = 0$ for all points in S_3 and $h_1(x) = 1$ for all points in S_2.

$w_{21} = -1, w_{12} = 0, b_1 = 3$

(ii) Next we set the parameters w_{21}, w_{22} and b_2 of the neuron labeled h_2 so that its output $h_2(x) = \phi(w_{21}x_1 + w_{22}x_2 + b_2)$ forms a linear separator between the sets S_1 and S_2.

(a) [1 pt.] On Fig 3b, draw a linear decision boundary that separates S_1 and S_2.

(b) [1 pt.] Write down the corresponding weights w_{21}, w_{22}, and b_2 so that $h_2(x) = 0$

for all points in S_1 and $h_2(x) = 1$ for all points in S_2.

$w_{11} = 3, w_{12} = 1, b_1 = 7$

(iii) Now we have two classifiers h_1 (to classify S_2 from S_3) and h_2 (to classify S_1 from S_2). We will set the weights of the final neuron of the neural network based on the results from h_1 and h_2 to classify the crosses from the circles. Let $h_3(x) = \phi\big(w_{31}h_1(x) + w_{32}h_2(x) + b_3\big)$.

(a) [1 pt.] Compute w_{31}, w_{32}, b_3 such that $h_3(x)$ correctly classifies the entire dataset. $w_{31} = 1, w_{32} = 1, b_3 = -1.5$

(b) [1 pt.] Draw your decision boundary in Fig 3c.

3. We would like to use neural networks classify the red points from blue points as shown in Fig.4a. The points in Fig.4a form three groups, from left to right, we denote them as $S1, S2$ and $S3$. Obviously these points are not linearly separable, you would like to use neural networks to classify the points. Define a layer of neural network in the same way as in your homework

$$y = \phi(w_1x_1 + w_2x_2 + b),$$

where the activation function is $\phi(x) = 1$ if $x > 0$ and $\phi(x) = 0$ if $x \leq 0$.

(i) To make the problem easier, we first consider set $S1$ and $S2$ in Fig. 4b. We use one layer neural network to classify $S1$ from $S2$. Define the classifier as $h_1(x) = \phi(w_{11}x_1 + w_{12}x_2 + b_1)$, find out w_{11}, w_{12}, b_1 such that $h_1(x) = 0 \quad \forall x \in S1$ and $h_1(x) = 1 \quad \forall x \in S2$. Draw the decision boundary in Fig. 4b.

$w_{11} = 2, w_{12} = 1, b_1 = 6$

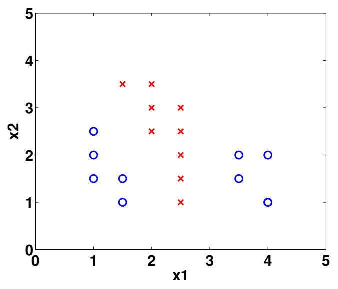
(ii) We then consider set $S2$ and $S3$ in Fig. 4c. We use one layer neural network to classify $S2$ from $S3$. Define the classifier as $h_2(x) = \phi(w_{21}x_1 + w_{22}x_2 + b_2)$, find out w_{21}, w_{22}, b_2 such that $h_2(x) = 0 \quad \forall x \in S3$ and $h_2(x) = 1 \quad \forall x \in S3$. Draw the decision boundary in Fig. 4c.

$w_{21} = 2, w_{22} = -1, b_2 = -4$

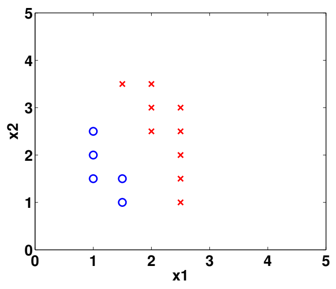
(iii) Now that we have two classifiers h_1, h_2 that can classify $S1$ from $S2$ and classify $S2$ from $S3$ separately, we can build another layer of neural network based on the results from h_1 and h_2 to classify $S2$ from $S1$ and $S3$. Let $h_3(x) = \phi(w_{31}h_1(x) + w_{32}h_2(x) + b_3)$, where h_1, h_2 are defined as in previous two steps. Find out w_{31}, w_{32}, b_3 such that $h_3(x) = 1 \quad \forall x \in S2$ and $h_3(x) = 0 \quad \forall x \in S1, S3$. Draw your decision boundary in Fig. 4a.

$w_{31} = 1, w_{32} = 1, b_3 = -1$

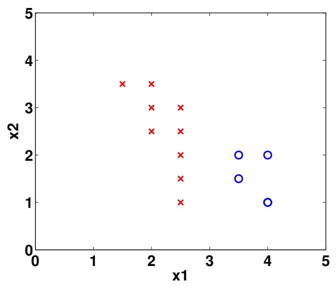
(iv) **Back propagation**
In the above example, we need to learn the weights by according to the data. At first step, we need to get the gradients of the parameters of neural networks.



(a) Sets $S1$, $S2$ and $S3$



(b) Subset S1 and S2



(c) Subset S2 and S3

Figure 4

Suppose there m data points x_i with label y_i, where $i \in [1, m]$. x_i is a $d \times 1$ vector and $y_i \in \{0, 1\}$. We use the data to train a neural network with one hidden layer:

$$h(x) = \sigma(W_1 x + b_1)$$
$$p(x) = \sigma(W_2 h(x) + b_2),$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$ is the sigmoid function, W_1 is a n by d matrix and b_1 is a n by 1 vector, W_2 is a 1 by n matrix and b_1 is a 1 by 1 vector.

We use cross entropy loss function and minimize the negative log likelihood to train the neural network:

$$l = \frac{1}{m} \sum_i -(y_i \log p_i + (1 - y_i) \log(1 - p_i)),$$

where $p_i = p(x_i), h_i = h(x_i)$.

(a) Describe how you would drive the gradients w.r.t the parameters W_1, W_2 and b_1, b_2. (No need to write out the detailed mathematical expression.) Use chain rule.

(b) When m is large, we typically use a small sample of all the data set to estimate the gradient, this is call stochastic gradient descent (SGD). Explain why we use SGD instead of gradient descent.
SGD converges faster than gradient descent.

(c) Work out the following gradient: $\frac{\partial l}{\partial p_i}, \frac{\partial l}{\partial W_2}, \frac{\partial l}{\partial b_2}, \frac{\partial l}{\partial h_i}, \frac{\partial l}{\partial W_1}, \frac{\partial l}{\partial b_1}$. When deriving the gradient w.r.t. the parameters in lower layers, you can may assume the gradient in upper layers are available to you (i.e., you can use them in your equation). For example, when calculating $\frac{\partial l}{\partial W_1}$, you can assume $\frac{\partial l}{\partial p_i}, \frac{\partial l}{\partial W_2}, \frac{\partial l}{\partial b_2}, \frac{\partial l}{\partial h_i}$ are known.

$$\begin{aligned}
\frac{\partial l}{\partial p_i} &= \frac{1}{m}\left(-\frac{y_i}{p_i} + \frac{1-y_i}{1-p_i}\right) \\
\frac{\partial l}{\partial W_2} &= \frac{1}{m}\sum_i \frac{\partial l_i}{\partial p_i}\frac{\partial p_i}{\partial W_2} = \frac{1}{m}\sum_i \frac{\partial l_i}{\partial p_i} p_i(1-p_i)h_i^T \\
\frac{\partial l}{\partial b_2} &= \frac{1}{m}\sum_i \frac{\partial l_i}{\partial p_i} p_i(1-p_i) \\
\frac{\partial l}{\partial h_i} &= \frac{\partial l_i}{\partial p_i}\frac{\partial p_i}{\partial h_i} = \frac{\partial l_i}{\partial p_i} p_i(1-p_i)W_2 \\
\frac{\partial l}{\partial W_1} &= \frac{1}{m}\sum_i \frac{\partial l_i}{\partial h_i}\frac{\partial h_i}{\partial W_2} = \frac{1}{m}\sum_i \frac{\partial l_i}{\partial h_i} h_i(1-h_i)x_i^T \\
\frac{\partial l}{\partial b_1} &= \frac{1}{m}\sum_i \frac{\partial l_i}{\partial h_i}\frac{\partial h_i}{\partial b_2} = \frac{1}{m}\sum_i \frac{\partial l_i}{\partial h_i} h_i(1-h_i)
\end{aligned}$$

4 PAC Learning

1. **True and Sample Errors:** Consider a classification problem with distribution D and target function $c^* : \mathcal{R}^d \mapsto \pm 1$. For any sample S drawn from D, answer whether the following statements are true or false, along with a brief explanation.

   a) [**4 pts**] For a given hypothesis space H, it is possible to define a sufficient size of S such that the true error is bounded by the sample error by a margin ϵ, for all hypotheses $h \in H$ with a given probability. True.

   b) [**4 pts**] The true error of any hypothesis h is an upper bound on its training error on the sample S. False. We said true error is close to training error, but it might be smaller than training error, so it is not an upper bound.

2. Let X be the feature space and there is a distribution D over X. We have training samples

$$S : (x_1, c^\star(x_1)), \cdots, ((x_m, c^\star(x_m))),$$

   x_i i.i.d from D. We assume labels $c^\star(x_i) \in \{-1, 1\}$.

   Let $\mathcal{H}$ be a concept class and let $h \in \mathcal{H}$ be a concept. In this question we restrict ourselves to $\mathcal{H}$. We use

$$err_S(h) = \frac{1}{m} \sum_{i=1}^{m} \mathbf{I}(h(x_i) \neq c^\star(x_i))$$

   to denote the training error and

$$err_D(h) = \mathbf{Pr}_{x \sim D}(h(x) \neq c^\star(x))$$

   to denote the true error. Recall the theorem from class, if the concept class is finite, in the realizable case

$$m \geq \frac{1}{\epsilon}\left[\ln(\mathcal{H}) + \ln\left(\frac{1}{\delta}\right)\right]$$

   labeled examples are sufficient so that with probability at least $1-\delta$, all $h \in \mathcal{H}$ with $err_D(h) \geq \epsilon$ have $err_S(h) > 0$; in the agnostic case,

$$m \geq \frac{1}{2\epsilon^2}\left[\ln(\mathcal{H}) + \ln\left(\frac{2}{\delta}\right)\right]$$

   labeled examples are sufficient such that with probability at least $1-\delta$, all $h \in \mathcal{H}$ have $err_D(h) - err_S(h) < \epsilon$.

   a) [**2 pts**] Briefly explain what is realizable case and what is agnostic case.

   Realizable- the true classifier, $c*$, is in $\mathcal{H}$
   Agnostic- $c* \notin \mathcal{H}$, but "close"

b) **[4 pts]** What is the full name of PAC learning? What is the correspondence between ϵ, δ and the full name?

"Probably approximately correct." The hypotheses we find with m examples are *probably* (with probability $p \geq 1 - \delta$) *approximately* correct, with $err_D(h) \leq \epsilon$

c) **[4 pts]** (True or False) Consider two concept finite classes $\mathcal{H}_1$ and $\mathcal{H}_2$ such that $\mathcal{H}_1 \subset \mathcal{H}_2$. Let $h_1 = \arg\min_{h \in \mathcal{H}_1} err_S\left(h\right)$ and $h_2 = \arg\min_{h \in \mathcal{H}_2} err_S\left(h\right)$. Thus according to the theorem, because $\mathcal{H}_2 \geq \mathcal{H}_1$, $err_D\left(h_2\right) \geq err_D\left(h_1\right)$. Briefly justify your answer.

False. Since there are more hypotheses in $\mathcal{H}_{\in}$ there might be one that better fits the data than those in $\mathcal{H}_{\infty}$.

5 MLE/MAP

1. Please circle **True** or **False** for the following questions, providing brief explanations to support your answer.

   (i) **[2 pts]** Consider the linear regression model $y = w^T x + \epsilon$. Assuming $\epsilon \sim \mathcal{N}(0, \sigma^2)$ and maximizing the conditional log-likelihood is equivalent to minimizing the sum of squared errors $\|y - w^T x\|_2^2$.

   Circle one: True False
   One line justification (only if False):

   True. The squared error term comes from the squared term in the Gaussian distribution.

   (ii) **[4 pts]** Consider n data points, each with one feature x_i and an output y_i. In linear regression, we assume $y_i \sim \mathcal{N}(wx_i, \sigma^2)$ and compute $\hat{w}$ through MLE.

   Suppose $y_i \sim \mathcal{N}(\log(wx_i), 1)$ instead. Then the maximum likelihood estimate $\hat{w}$ is the solution to the following equality:

$$\sum_{i=1}^{n} x_i y_i = \sum_{i=1}^{n} x_i \log(wx_i)$$

   Circle one: **True** **False**
   Brief explanation:

   False. The likelihood function can be written as

$$\prod_{i=1}^{n} \frac{\exp(-(y_i - \log(wx_i))^2/2)}{\sqrt{2\pi}} = \frac{\exp(-\sum_{i=1}^{n}(y_i - \log(wx_i))^2/2)}{\sqrt{2\pi}}$$

   Differentiating wrt w and setting to zero gives us

$$\sum_{i=1}^{n} 2(y_i - \log(wx_i))\frac{x_i}{wx_i} = 0 \implies \sum_{i=1}^{n} y_i = \sum_{i=1}^{n} \log(wx_i)$$

2. **Select all that apply:** Which of the following are correct regarding Gradient Descent (GD). Assume data log-likelihood is $L(\theta|X)$, which is a function of the parameter θ, and the objective function is negative log-likelihood .

   - ☐ GD requires that $L(\theta|X)$ is concave with respect to parameter θ in order to converge

- ☐ GD requires that $L(\theta|X)$ is convex with respect to parameter θ in order to converge
- ☐ GD update rule is $\theta \leftarrow \theta - \alpha\nabla_\theta L(\theta|X)$
- ☐ Given a fixed small learning rate (say $\alpha = 10^{-10}$), GD will always reach the optimum after infinite iterations (assume that the objective function satisfies the convergence condition).

A

Analysis: C should replace minus with plus. D is wrong because it is possible that θ will jump around the minimum and never reach the optimum even though α is (finitely) small.

3. Let $X_1, X_2, ..., X_N$ be i.i.d. data from a uniform distribution over a diamond-shaped area with edge length $\sqrt{2}\theta$ in $\mathbb{R}^2$, where $\theta \in \mathbb{R}^+$ (see Figure 5). Thus, $X_i \in \mathbb{R}^2$ and the distribution is

$$p(x|\theta) = \begin{cases} \frac{1}{2\theta^2} & if \ \|x\| \leq \theta \\ 0 & otherwise \end{cases}$$

where $\|x\| = |x_1| + |x_2|$ is $L1$ norm. Please find the maximum likelihood estimator of θ.

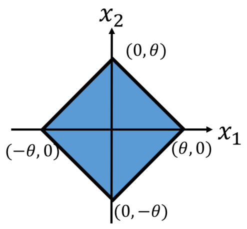


Figure 5: Area of $\|x\| \leq \theta$

Analysis:

The likelihood function is

$$L(X_1, X_2, ..., X_N; \theta) = \frac{1}{(2\theta^2)^N} 1\left\{\underset{1\leq i\leq N}{max} \|X_i\| \leq \theta\right\}$$

To maximize likelihood, we want θ to be as small as possible with the constraint of $\underset{1\leq i\leq N}{max} \|X_i\| \leq \theta$, otherwise the likelihood drops to 0. So the MLE of θ is

$$\widehat{\theta} = \underset{1\leq i\leq N}{max} \|X_i\|$$

4. **Short answer:** Suppose we want to model a 1-dimensional dataset of N real valued features $\left(x^{(i)}\right)$ and targets $\left(y^{(i)}\right)$ by:

$$y^{(i)} \sim \mathcal{N}\left(\exp(wx^{(i)}), 1\right)$$

Where w is our unknown (scalar) parameter and $\mathcal{N}$ is the normal distribution with probability density function:

$$f(a)_{\mathcal{N}(\mu,\sigma^2)} = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(a-\mu)^2}{2\sigma^2}\right)$$

Can the maximum conditional negative log likelihood estimator of w be solved analytically? If so, find the expression for w_{MLE}. If not, say so and write down the update rule for w in gradient descent.

Cannot be found analytically.

Taking the derivative of the negative log likelihood with respect to w yields:

$$\frac{\partial \text{NLL}}{\partial w} = \frac{\sum_i^N -2x^{(i)}y^{(i)}\exp(wx^{(i)}) + x^{(i)}\exp(2wx^{(i)})}{2}$$

Update rule is thus

$$w \leftarrow w - \eta \frac{\partial \text{NLL}}{\partial w}$$

5. Assume we have n iid random variables $x_i, i \in [1,n]$ such that each x_i belongs to a normal distribution with mean μ and variance σ^2.

$$p(x_1, x_2, ..., x_n|\mu, \sigma^2) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} exp\{\frac{-(x_i-\mu)^2}{2\sigma^2}\}$$

a) Write the log likelihood function $l(x_1, x_2...x_n|\mu, \sigma^2)$

$$log(\prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} exp\{\frac{-(x_i-\mu)^2}{2\sigma^2}\}) = \sum_{i=1}^{n}[\log(\frac{1}{\sqrt{2\pi}\sigma}) - \frac{(x_i-\mu)^2}{2\sigma^2}] \tag{1}$$

$$= -n\log(\sqrt{2\pi}\sigma) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i-\mu)^2 \tag{2}$$

b) Derive an expression for the Maximum Likelihood Estimate for the variance (σ^2)

We can find the estimator by solving $\nabla_\sigma l(x_1, x_2...x_n|\mu, \sigma^2) = 0$.

$$-n\frac{1}{\sqrt{2\pi}\sigma}\sqrt{2\pi} + \frac{1}{\sigma^3}\sum_{i=1}^{n}(x_i-\mu)^2 = 0 \tag{3}$$

$$\frac{1}{\sigma^2}\sum_{i=1}^{n}(x_i-\mu)^2 = \frac{n}{\sigma} \tag{4}$$

$$\frac{1}{n}\sum_{i=1}^{n}(x_i-\mu)^2 = \sigma^2 \tag{5}$$

6. Assume we have a random variable that is Bernoulli distributed $X_1, \ldots, X_n \sim Bernoulli(\theta)$. We are going to derive its MLE. Recall that in a Bernoulli $X = \{0, 1\}$ and the pdf of a Bernoulli is

$$p(X;\theta) = \theta^x(1-\theta)^{1-x}$$

a) Derive the likelihood, $L(\theta; X_1, \ldots, X_n)$

$L(\theta; X_1, \ldots, X_n) = \prod_{i=1}^n p(X_i;\theta)$
$L(\theta; X_1, \ldots, X_n) = \prod_{i=1}^n \theta^{x_i}(1-\theta)^{1-x_i}$
$L(\theta; X_1, \ldots, X_n) = \theta^{\sum_{i=1}^n x_i}(1-\theta)^{n-\sum_{i=1}^n x_i}$
Either of the final two steps are acceptable.

b) Derive the following formula for the log likelihood

$$l(\theta; X_1, \ldots, X_n) = (\sum_{i=1}^n X_i)\log(\theta) + (n - \sum_{i=1}^n X_i)\log(1-\theta)$$

$l(\theta; X_1, \ldots, X_n) = \log L(\theta; X_1, \ldots, X_n)$
$l(\theta; X_1, \ldots, X_n) = \log \left[\theta^{\sum_{i=1}^n x_i}(1-\theta)^{n-\sum_{i=1}^n x_i}\right]$
$l(\theta; X_1, \ldots, X_n) = (\sum_{i=1}^n x_i)\log(\theta) + (n - \sum_{i=1}^n x_i)\log(1-\theta)$

c) Derive the MLE, $\hat{\theta}$, and show that $\hat{\theta} = \frac{1}{n}(\sum_{i=1}^n X_i)$

Take the derivative of the log likelihood and set it to zero
$\frac{dl}{d\theta} = \frac{d}{d\theta}\left[(\sum_{i=1}^n x_i)\log(\theta) + (n - \sum_{i=1}^n x_i)\log(1-\theta)\right] = 0$
$\frac{\sum_{i=1}^n x_i}{\theta} - \frac{n - \sum_{i=1}^n x_i}{1-\theta} = 0$
$\left(\sum_{i=1}^n x_i\right)(1-\theta) - \left(n - \sum_{i=1}^n x_i\right)\theta = 0$
$\sum_{i=1}^n x_i - n\theta = 0$
$\hat{\theta} = \frac{1}{n}(\sum_{i=1}^n X_i)$

7. Assume we have a random sample that is Bernoulli distributed $X_1, \ldots, X_n \sim \text{Exponential}(\theta)$. We are going to derive the MLE for θ. Recall that a exponential random variable X has p.d.f:

$$P(X;\theta) = \theta\exp(-\theta X).$$

a) Derive the likelihood, $L(\theta; X_1, \ldots, X_n)$.

$L(\theta; X_1, \ldots, X_n) = \prod_{i=1}^n p(X_i;\theta)$
$L(\theta; X_1, \ldots, X_n) = \prod_{i=1}^n \theta\exp^{-\theta x_i}$

b) Find θ that maximizes $L(\theta; X_1, \ldots, X_n)$.

$l(\theta; X_1, \ldots, X_n) = \log L(\theta; X_1, \ldots, X_n)$
$l(\theta; X_1, \ldots, X_n) = \log\left[\prod_{i=1}^n \theta\exp^{-\theta x_i}\right]$

$l(\theta; X_1, \ldots, X_n) = \sum_{i=1}^n log(\theta) + \sum_{i=1}^n -\theta x_i$
$\frac{dl}{d\theta} = \frac{d}{d\theta}\left[\sum_{i=1}^n log(\theta) + \sum_{i=1}^n -\theta x_i\right] = 0$
$\sum_{i=1}^n \frac{1}{\theta} - \sum_{i=1}^n x_i = 0$

$$\theta = \frac{n}{\sum_{i=1}^n x_i}$$

8. For each question state **True** or **False** and give one line justifications.

a) **T or F** The value of the Maximum Likelihood Estimate (MLE) is equal to the value of the Maximum A Posteriori (MAP) Estimate with a uniform prior.

True. Since we know posterior is proportional to the product of likelihood and prior, i.e.,

$$p(\theta|x) \propto p(x|\theta)p(\theta). \tag{6}$$

Since uniform prior gives us constant value on $p(\theta)$, after proper normalization, we know likelihood and posterior are the same, so do their estimator.

b) **T or F** The bias of the Maximum Likelihood Estimate (MLE) is typically less than or equal to the bias of the Maximum A Posteriori (MAP) Estimate.

True. Since MAP estimate allows us to incorporate more prior knowledge, it is likely to be more biased.

c) **T or F** The MAP estimate is always better than the MLE.

False. When the prior is chosen poorly, it will take the MAP estimate longer to converge to a good estimate because it needs to see enough examples to overcome the bad prior.

d) **T or F** In the limit as n (the number of samples) increases, the MAP and MLE estimates become the same.

True. As the number of examples increases, the data likelihood goes to zero very quickly, while the magnitude of the prior stays the same. In the limit, the prior plays an insignificant role in the MAP estimate and the two estimates will converge.

e) **T or F** Naive Bayes can only be used with MAP estimates, and not MLE estimates.

False. Naive Bayes can be used with any technique for estimating the parmaters of a distribution. In homework 2, we used both the MAP and MLE estimates.

6 Probability, Naive Bayes and MLE

6.1 Probability

1. For each question, circle the correct option.

   1. Which of the following expressions is equivalent to $p(A|B,C,D)$?

      (a) $\frac{p(A,B,C,D)}{p(C|B,D)p(B|D)p(D)}$

      (b) $\frac{p(A,B,C,D)}{p(B,C)p(D)}$

      (c) $\frac{p(A,B,C,D)}{p(B,C|D)p(B)p(C)}$

      Answer is (a). $p(A|BCD) = \frac{p(A,B,C,D)}{p(B,C,D)} = \frac{p(A,B,C,D)}{p(C|B,D)p(B,D)} = \frac{p(A,B,C,D)}{p(C|B,D)p(B|D)p(D)}$

   2. Let μ be the mean of some probability distribution. $p(\mu)$ is always non-zero.

      (a) True

      (b) False

      No, taking the example of a distribution that put 0.5 probability on +1 and 0.5 probability on -1. Though their mean is 0, p(0) is zero.

2. Assume we have a sample space Ω. Just state **T** or **F**, no justification needed.

   1. If events A, B, and C are disjoint then they are independent.

      False. If they are disjoint, i.e. mutually exclusive, they are very dependent! (what does disjoint mean in terms of the probabilities of A, B, and C? What about independent?)

   2. $P(A|B) \propto \dfrac{P(A)P(B|A)}{P(A|B)}$.

      False $P(A|B) \propto \dfrac{P(A)P(B|A)}{P(B)}$

   3. $P(A \cup B) \leq P(A)$.

      False $P(A \cup B) \geq P(A)$

   4. $P(A \cap B) \geq P(A)$.

      False $P(A \cap B) \leq P(A)$

6.2 Naive Bayes

1. Consider the following data. It has 4 features $\mathbf{X} = (x_1, x_2, x_3, x_4)$ and 3 labels $(+1, 0, -1)$. Assume that the probabilities $p(\mathbf{X}|y)$ and $p(y)$ are both Bernoulli distributions. Answer the questions that follow under the Naive Bayes assumption.

x_1	x_2	x_3	x_4	y
1	1	0	1	+1
0	1	1	0	+1
1	0	1	1	0
0	1	1	1	0
0	1	0	0	-1
1	0	0	1	-1
0	0	1	1	-1

1. Compute the Maximum Likelihood Estimate for $p(x_i = 1|y), \forall i \in [1,4], \forall y \in \{+1, 0, -1\}$.

	$y = +1$	$y = 0$	$y = -1$
$x_1 = 1$	0.5	0.5	1/3
$x_2 = 1$	1	0.5	1/3
$x_3 = 1$	0.5	1	1/3
$x_4 = 1$	0.5	1	2/3

2. Compute the Maximum Likelihood Estimate for the prior probabilities $p(y = +1), p(y = 0), p(y = -1)$

   $p(y = +1) = \frac{2}{7}$, $p(y = 0) = \frac{2}{7}$ and $p(y = -1) = \frac{3}{7}$.

3. Use the values computed in the above two parts to classify the data point $(x_1 = 1, x_2 = 1, x_3 = 1, x_4 = 1)$ as either belonging to class $+1, 0$ or -1

   According to naive bayes assumption, samples are independent given y, thus we can write the conditional joint probability as

$$p(x_1 = 1, x_2 = 1, x_3 = 1, x_4 = 1) = p(x_1 = 1, x_2 = 1, x_3 = 1, x_4 = 1|y)p(y) \quad (7)$$

$$= p(y) \prod_{i=1}^{4} p(x_i = 1|y). \quad (8)$$

   We calculate the probability given different value on y and pick the y that gives us largest probability.

$$p(y = +1) \prod_{i=1}^{4} p(x_i = 1|y = +1) = \frac{1}{2} \cdot 1 \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{2}{7} = \frac{1}{28} \quad (9)$$

$$p(y = 0) \prod_{i=1}^{4} p(x_i = 1|y = 0) = \frac{1}{2} \cdot \frac{1}{2} \cdot 1 \cdot 1 \cdot \frac{2}{7} = \frac{1}{14} \quad (10)$$

$$p(y = -1) \prod_{i=1}^{4} p(x_i = 1|y = -1) = \frac{1}{3} \cdot \frac{1}{3} \cdot \frac{1}{3} \cdot \frac{2}{3} \cdot \frac{3}{7} = \frac{2}{189} \quad (11)$$

Since $y = 0$ yields the largest value, we classify the data as $\hat{y} = 0$.

2. You are given a data set of 10,000 students with their sex, height, and hair color. You are trying to build a machine learning classifier to predict the sex of a student, so you randomly split the data into a training set and a testing set. Here are the specifications of the data set:

   - sex $\in$ {male,female}
   - height $\in$ [0,300] centimeters
   - hair $\in$ {brown, black, blond, red, green}
   - 3240 men in the data set
   - 6760 women in the data set

   Under the assumptions necessary for Naive Bayes (not the distributional assumptions you might naturally or intuitively make about the dataset) answer each question with **T** or **F** and a one sentence explanation of your answer:

   1. **T or F:** Height is a continuous valued variable. Therefore Naive Bayes is not appropriate since it cannot handle continuous valued variables.

      False. Naive Bayes can handle both continuous and discrete values as long as the appropriate distributions are used for conditional probabilities. For example, Gaussian for continuous and Bernoulli for discrete

   2. **T or F:** Since there is not a similar number of men and women in that dataset Naive Bayes will have high test error.

      False. Since the data was randomly split, the same proportion of male and female will be in the training and testing sets. Thus this discrepancy will not affect testing error.

   3. **T or F:** $p(\texttt{height}|\texttt{sex}, \texttt{hair}) = p(\texttt{height}|\texttt{sex})$.

      True. This results from the conditional independence assumption required for Naive Bayes.

   4. **T or F:** $p(\texttt{height}, \texttt{hair}|\texttt{sex}) = p(\texttt{height}|\texttt{sex}) * p(\texttt{hair}|\texttt{sex})$.

      True. This results from the conditional independence assumption required for Naive Bayes.

6.3 Naive Bayes, Logistic Regression

1. Suppose you wish to learn $P(Y|X_1, X_2, X_3)$, where Y, X_1, X_2 and X_3 are all boolean-valued random variables. You consider both Naive Bayes and Logistic Regression as possible approaches.

   For each of the following, answer True or False, and give a *one sentence* justification for your answer.

1. T or F: In this case, a good choice for Naive Bayes would be to implement a Gaussian Naive Bayes classifier.

   Answer: False. Given the X_i are boolean, it is better to model $P(X_i|Y)$ with a Bernoulli rather than Gaussian distribution

2. T or F: To learn $P(Y|X_1, X_2, X_3)$ using Naive Bayes, you must make conditional independence assumptions, including the assumption that Y is conditionally independent of X_1 given X_2.

   Answer: False. Naive Bayes assume $(\forall i \neq j) X_i$ is conditionally independent of X_j given Y.

3. T or F: Logistic regression is certain to be the better choice in this case.

   Answer: False. It will depend on the number of training examples available, and whether the Naive Bayes assumptions are satisfied.

2. Parameter estimation

   1. How many parameters must be estimated for your Gaussian Naive Bayes classifier, and what are they (i.e., please list them).

      7 - We need $P(Y = 1)$ and $P(Y = 0) = 1 - P(Y = 1)$. Then we need $P(X_i = 0|Y = 0), P(X_i = 1|Y = 0), P(X_i = 0|Y = 1), P(X_i = 1|Y = 1)$ for $i \in \{1, 2, 3\}$.

      But given the binary parameters, $P(X_i = 1|Y = 0) + P(X_i = 0|Y = 0) = 1$, so we only need to estimate one of these conditional probabilities ($P(X_i = 0|Y = 0)$, for example). So in total we need only 2*3 + 1 = 7

   2. How many parameters must be estimated for your Logistic Regression classifier, and what are they (i.e., please list them).

      4 - Weights for: Bias, X_1, X_2, and X_3

   3. T or F: We can train Naive Bayes using maximum likelihood estimates for each parameter, but not MAP estimates. Justify your answer *in one sentence.*

      False: MAP estimates are just MLE spiced up with priors on the parameters of $P(X_i|Y_j)$ (prior knowledge that we can inject into the model), so there's no reason we can't add it in.

   4. T or F: We can train Logistic Regression using maximum likelihood estimates for each parameter, but not MAP estimates. Justify your answer *in one sentence.*

      False: We can assign priors on the regression model by assuming $y = w^T x + \epsilon$ where ϵ is "noise" from a distribution (e.g. Gaussian)

3. Mixing discrete and continuous variables. Suppose we add a numeric, real-valued variable X_4 to our problem. Note we now have a mix of some discrete-valued X_i and one continuous X_i.

1. Explain in *two sentences* why we can no longer use Naive Bayes, or if we can, how we would modify our first solution.

   We can't use our original model because a Bernoulli distribution can't model the new data.

   We must modify our solution so that a different Naive Bayes model is trained on the continuous variables, using a different distribution than a Bernoulli one (i.e. a Gaussian), and then the result is a multiplication of the two.

   Since our assumption is that each parameter is conditionally independent anyhow, we can multiply the results of the two models together safely.

2. Explain in *two sentences* why we can no longer use Logistic Regression, or if we can, how we would modify our first solution.

   Assuming the discrete variables were already transformed into one-hot features, we can simply add one weight per continuous feature to our model.