6.867: Exercises (Week 8)

November 3, 2017

Contents

Solution: Don't look at the solutions until you have tried your absolute hardest to solve the problems.

1 Money tree [1]

You have a decision tree algorithm and you are trying to figure out which attribute is the best to test on first. You are using the information gain metric.

- You are given a set of 128 examples, with 64 positively labeled and 64 negatively labeled.
- There are three attributes, Home Owner, In Debt, and Rich.
- For 64 examples, Home Owner is true. The Home Owner=true examples are 1/4 negative and 3/4 positve.
- For 96 examples, In Debt is true. Of the In Debt=true examples, 1/2 are positive and half are negative.
- For 32 examples, Rich is true. 3/4 of the Rich=true examples are positive and 1/4 are negative.

Use $\log_2$ in your computations.

(a) What is the entropy of the initial set of examples?

Solution: 1.0

(b) What is the information gain of splitting on the Home Owner attribute as the root node?

Solution: Entropy(0.25, 0.75) = 0.811
Gain = 1 - (64/128)*(0.811) - (64/128)*(0.811) = 0.188

(c) What is the information gain of splitting on the In Debt attribute as the root node?

Solution: Entropy(0.5, 0.5) = 1
Gain = 1 - (96/128)*(1) - (32/128)*(1) = 0

(d) What is the information gain of splitting on the Rich attribute as the root node?

Solution: Entropy(0.25, 0.75) = 0.811
Entropy(0.41, 0.59) = 0.98
Gain = 1 - (32/128)*(0.811) - (96/128)*(0.98) = 0.062

[1] CMU 15-381 Fall 2001 Homework 5, Problem 2

(e) Which attribute do you split on?

Solution: Home Owner, which has the highest information gain.

2 Gradient boosting

Boosting is a method for making an additive model, where we explicitly construct new data sets for training each new member of the ensemble, so that new classifiers attempt to "make up for" weaknesses of the current committee.

Assuming we have a binary classification problem with data points (x, y), and we have an ensemble of classifiers $h(x;\theta)$, where θ is the parameter of the classifier. When using m classifiers with weights α, the prediction is given by the sign of the weighted sum of individual classifier outputs

$$h_m(x) = \sum_{j=1}^{m} \alpha_j h(x;\theta_j)$$

where $\alpha_j \geqslant 0$ (not necessarily summing to one) is the weight for classifier j. The base learners $h(x;\theta_j)$ are often simple (weak) classifiers such as decision stumps. There are many variations on the boosting algorithm due to the choice of base learners or the loss function that the overall algorithm seeks to minimize.

The algorithm starts with no learners $h_0(x) = 0$, then at each stage of the algorithm, we add a weak learner to the current ensemble of learners to minimize the overall loss. Therefore, at stage m of the algorithm, we will try to minimize

$$J(\alpha_m, \theta_m) = \sum_{i=1}^{n} \text{Loss}\Big(y^i h_m(x^i)\Big) = \sum_{i=1}^{n} \text{Loss}\Big(y^i h_{m-1}(x^i) + y^i \alpha_m h(x^i;\theta_m)\Big)$$

where we assume that $h_{m-1}(x)$ is fixed from $m-1$ previous rounds and view J as a function of parameters α_m and θ_m associated with the new base classifier. The margin loss $\text{Loss}(\cdot)$ could be any function that is convex and decreasing in its argument (smaller loss for predictions with larger voting margin).

Let's elaborate on the algorithm a bit. We can initialize $h_0(x) = 0$ for the ensemble with no base learners. Then, at the m-th round, we perform two distinct steps to optimize θ_m and α_m, respectively. First, we find the best new classifier $h(x;\hat{\theta}_m)$ that helps with the current ensemble. Thus, we find the parameters $\hat{\theta}_m$ that minimize

$$\frac{\partial}{\partial \alpha_m} J(\alpha_m, \theta_m)\Big|_{\alpha_m=0} = \sum_{i=1}^{n} \overbrace{dL\Big(y^i h_{m-1}(x^i)\Big)}^{-W_{m-1}(i)} y^i h(x^i;\theta_m) = -\sum_{i=1}^{n} W_{m-1}(i)\, y^i h(x^i;\theta_m)$$

where $dL(z) = d/dz\, \text{Loss}(z)$ is always negative because the loss is decreasing. (We applied the chain rule of derivatives to get the above form). If we consider $W_{m-1}(i)$ as the weights of sample i after $m-1$ rounds, then we are effectively looking for a classifier that minimizes the weighted

classification loss on the dataset. The values of the sample weights will differ based on different loss functions but they will always decrease as a function of how well the current ensemble classifies the training examples. We can also normalize these weights in the algorithm since the normalization does not affect the resulting $\hat{\theta}_m$.

Once we have $\hat{\theta}_m$, we determine the weight $\hat{\alpha}_m$ of the new classifier $h(x;\theta_m)$ that minimizes

$$J(\alpha_m, \hat{\theta}_m) = \sum_{i=1}^{n} \text{Loss}\left(y^i h_{m-1}(x^i) + y^i \alpha_m h(x^i; \hat{\theta}_m)\right)$$

We usually will not get a closed-form expression for $\hat{\alpha}_m$. However, the optimization problem is easy to solve since $\text{Loss}(\cdot)$ is convex and we only have one parameter to optimize.

At the end of each round, we reweight the weights of each data sample so that the classifiers added in the future focuses more on the misclassified samples.

We can now write the general boosting algorithm more succinctly. After initializing $W_0(i) = 1/n$, each boosting iteration consists of the following three steps:

(Step 1) Find $\hat{\theta}_m$ that (approximately) minimizes the weighted error ϵ_m or $2\epsilon_m - 1$ given by

$$-\sum_{i=1}^{n} W_{m-1}(i)\, y^i h(x^i; \theta_m)$$

(Step 2) Find $\hat{\alpha}_m$ that minimizes

$$J(\alpha_m, \hat{\theta}_m) = \sum_{i=1}^{n} \text{Loss}\left(y^i h_{m-1}(x^i) + y^i \alpha_m h(x^i; \hat{\theta}_m)\right)$$

(Step 3) Reweight the examples

$$W_m(i) = -c_m\, dL\left(y^i h_{m-1}(x^i) + y^i \hat{\alpha}_m h(x^i; \hat{\theta}_m\right)$$

where c_m normalizes the new weights so that they sum to one.

Now that we have a boosting algorithm for any loss function, we can select a particular one. Specifically, we will consider the logistic loss:

$$\text{Loss}(z) = \log\left(1 + \exp(-z)\right)$$

(a) Show that the unnormalized weights $W_m(i)$ from the logistic loss are bounded by 1. What can you say about the resulting normalized weights for examples that are clearly misclassified in comparison to those that are just slightly misclassified by the current ensemble? If the training data contains mislabeled examples, why do we prefer the logistic loss over the exponential loss, $\text{Loss}(z) = \exp(-z)$?

Solution:

Compute the negative derivative of the logistic loss

$$W_m(t) = -\frac{dL(z)}{dz} = \frac{\exp(-z)}{1+\exp(-z)} = \frac{1}{\exp(z)+1} < 1$$

For clearly misclassified examples, $y^t h_m(x^t)$ is a large negative, so $W_m(t)$ is close to 1, while for slightly misclassifed examples, $y^t h_m(x^t)$ is a small negative, so $W_m(t)$ is close to $\frac{1}{2}$. Thus, the normalized weights for the two respective cases will be in a ratio of at most 2:1, i.e., a single clearly misclassifed outlier will never be worth more than two completely uncertain points. This is why boosting with logistic loss is more robust to outliers that with exponential loss.

(b) Suppose the training set is linearly separable and we would use a hard-margin linear support vector machine (no slack) as a base classifier. In the first boosting iteration, what would the resulting $\hat{\alpha}_1$ be?

Solution: In the first step, since the dataset is linearly separable, the hard margin SVM would produce a separator that has $y^t h(x^t; \theta_1) \geqslant 1$ for all t. Then, if we pick α_1 to minimize

$$J(\alpha_1, \theta_1) = \sum_{t=1}^{n} L(\alpha_1 y^t h_1(x^t; \theta_1))$$

it is clear that we need to strictly increase α_1 because $y^t h_1(x^t; \theta_1) \geqslant 1$. Therefore, the boosting algorithm will set $\alpha = \infty$. This makes sense, because if we can find a base classifier that perfectly separates the data, we will weight it as much as possible to minimize the boosting loss.

(c) Show that we need at most $2n$ stumps to correctly classify n training examples with distinct coordinate values. Consider the case in which the training examples are distinct points in one dimension.

Solution: Consider any n distinct points $(x^i : i = 1, ..., n)$ in one dimension. As the points are distinct and n is finite, there exists a positive real $\epsilon > 0$, such that $N_\epsilon(x^i) = (x^i - \epsilon, x^i + \epsilon)$, i.e., the open interval of radius ϵ around each point x^i. Clearly, $N_\epsilon(x^i)$ are all disjoint. Then, for each x^i, consider the two stumps $f_i^+ = sign(y^i(x - (x^i - \epsilon)))$ and $f_i^- = sign(-y^i(x - (x^i + \epsilon)))$. These two stumps perfectly classifies x^i while leaving the other points alone because

$$F_i(x) = \frac{1}{2}f_i^+(x) + \frac{1}{2}f_i^-(x) = y^i, x \in N_\epsilon(x^i) \tag{2.1}$$

$$F_i(x) = \frac{1}{2}f_i^+(x) + \frac{1}{2}f_i^-(x) = 0, x \notin N_\epsilon(x^i) \tag{2.2}$$

It follows that $F(x) = \sum_{i=1}^{n} F_i(x)$ is an ensemble of $2n$ stumps that allow us to perfectly classify any n distinct points in 1 dimension.

3 Holmes and Watson in LA

Holmes and Watson have moved to LA. Holmes wakes up to find that his lawn is wet. He wonders if it has rained or if he left his sprinkler on. He looks at his neighbor Watson's lawn and sees that it is wet, too. So, he concludes it must have rained.

Use the binary random variables R for rain, S for sprinkler, H for Holmes' lawn being wet and W for Watson's lawn being wet. Assume you are given the following probability distributions:

$$\begin{aligned}
P(R = 1) &= 0.2 \\
P(S = 1) &= 0.1 \\
P(W = 1 \mid R = 0) &= 0.2 \\
P(W = 1 \mid R = 1) &= 1.0 \\
P(H = 1 \mid R = 0, S = 0) &= 0.1 \\
P(H = 1 \mid R = 0, S = 1) &= 0.9 \\
P(H = 1 \mid R = 1, S = 0) &= 1.0 \\
P(H = 1 \mid R = 1, S = 1) &= 1.0
\end{aligned}$$

(a) Draw the corresponding directed graphical model?

(b) What is $P(H)$?

(c) What probability expression corresponds to Holmes' belief that it rained before he goes out? What is its value?

(d) What probability expression corresponds to Holmes' belief that it rained after he sees that his lawn is wet? What is its value?

(e) What probability expression corresponds to Holmes' belief that it rained after he sees that his lawn is wet and that the sprinkler is off? What is its value?

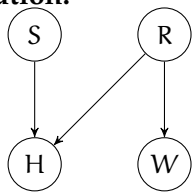
(f) What probability expression corresponds to Holmes' belief that it rained after he sees that his lawn is wet and that Watson's is wet as well? What is its value?

(g) What probability expression corresponds to Holmes' belief that the sprinkler was on after he sees that his lawn is wet and that Watson's is wet as well? What is its value?

(h) Discuss the results of questions (c)-(e). What does these results suggest about conditional dependency of S and R (conditioned on H)?

Solution:

(a)



(b) We have $P = P(R, S, W, H) = P(R)P(S)P(W|R)P(H|R, S)$, thus

$$P(H = 1) = \sum_{R,S,W} P(R)P(S)P(W|R)P(H = 1|R, S) \tag{3.1}$$

$$= 0.344 \tag{3.2}$$

(c)

$$P(R = 1) = 0.2 \tag{3.3}$$

(d)

$$P(R = 1|H = 1) = \frac{\sum_{S,W} P(R = 1)P(S)P(W|R = 1)P(H = 1|R = 1, S)}{\sum_{R,S,W} P(R)P(S)P(W|R)P(H = 1|R, S)} \tag{3.4}$$

$$= \frac{0.2}{0.2 + 0.144} = 0.581 \tag{3.5}$$

(e)

$$P(R = 1|H = 1, S = 0) = \frac{\sum_{W} P(R = 1)P(S = 0)P(W|R = 1)P(H = 1|R = 1, S = 0)}{\sum_{R,W} P(R)P(S = 0)P(W|R)P(H = 1|R, S = 0)} \tag{3.6}$$

$$= \frac{0.18}{0.18 + 0.072} = 0.714 \tag{3.7}$$

(f)

$$P(R = 1|H = 1, W = 1) = \frac{\sum_{S} P(R = 1)P(S)P(W = 1|R = 1)P(H = 1|R = 1, S)}{\sum_{R,S} P(R)P(S)P(W = 1|R)P(H = 1|R, S)} \tag{3.8}$$

$$= \frac{0.2}{0.2 + 0.0288} = 0.874 \tag{3.9}$$

(g)

$$P(S = 1|H = 1, W = 1) = \frac{\sum_{R} P(R)P(S = 1)P(W = 1|R)P(H = 1|R, S = 1)}{\sum_{R,S} P(R)P(S)P(W = 1|R)P(H = 1|R, S)} \tag{3.10}$$

$$= \frac{0.0344}{0.0344 + 0.1944} = 0.153 \tag{3.11}$$

(h) The probability $p(R = 1|H = 1)$ is higher than $p(R = 1)$, reflecting the fact that the observation of Holmes' lawn has some information about the weather. If we observe $S = 0$, the probability $p(R = 1|H = 1, S = 0)$ is even higher since the weather and the sprinkler are the two possible cause of wet lawn. Notice that this means when conditioned on H, R and S are no longer independent of each other.

4 Bayesian networks must be acyclic

Suppose we have a graph $G = (V, E)$ and discrete random variables $X_1, \ldots, X_n$, and define

$$f(x_1, \ldots, x_n) = \prod_{v \in V} f_v(x_v | x_{pa(v)})$$

where $pa(v)$ refers to the parents of variable X_v in G and $f_v(x_v|x_{pa(v)})$ specifies a distribution over X_v for every assignment to X_v's parents, i.e. $0 \leqslant f_v(x_v|x_{pa(v)}) \leqslant 1$ for all $x_v \in Vals(X_v)$ and $\sum_{x_v \in Vals(X_v)} f_v(x_v|x_{pa(v)}) = 1$. Recall that this is precisely the definition of the joint probability distribution associated with the Bayesian network G, where the f_v are the conditional probability distributions.

Show that if G has a directed cycle, f may no longer define a valid probability distribution. In particular, give an example of a cycle graph G and distributions f_v such that $\sum_{x_1,\ldots,x_n} f(x_1, \ldots, x_n) \neq 1$ (A valid probability distribution must be non-negative and sum to one). This is why Bayesian networks must be defined on *acyclic* graphs.

Solution: For an example of cyclic graph, we can consider a graph $X_1 \to X_2 \to X_3 \to X_1$. For simplicity we assume that all three variables X_1, X_2, X_3 are binary variables. Suppose that the conditional distribution is given by $f_v(x_v|x_{pa(v)}) = 1$ when $x_v = x_{pa(v)}$, and 0 otherwise. Then we know that only $(0, 0, 0)$ and $(1, 1, 1)$ have positive probabilities. Therefore,

$$\sum_{x_1,x_2,x_3} f(x_1, x_2, x_3) = f(0, 0, 0) + f(1, 1, 1) = 2 \neq 1$$

which violates the normalization property of probability distribution.

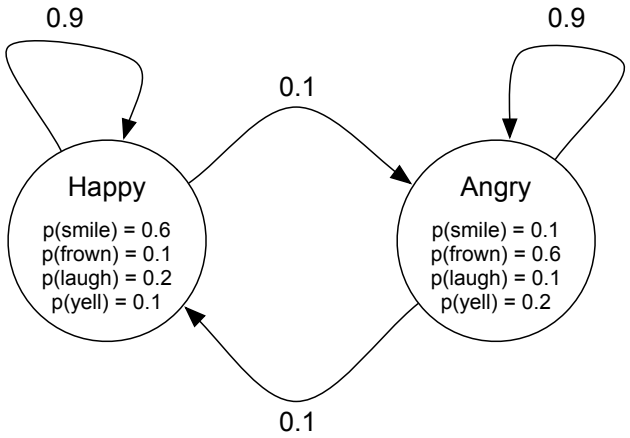
5 Hidden Markov Model

Harry lives a simple life. Some days he is Angry and some days he is Happy. But he hides his emotional state, and so all we can observe is whether he smiles, frowns, laughs, or yells. Harry's best friend is utterly confused about whether Harry is actually happy or angry and decides to model his emotional state using a hidden Markov model.

Let $X_d \in$ {Happy, Angry} denote Harry's emotional state on day d, and let $Y_d \in$ {smile, frown, laugh, yell} denote the observation made about Harry on day d. **Assume that on day 1 Harry is in the Happy state**, i.e. $X_1 =$ Happy. Furthermore, assume that Harry transitions between

states exactly once per day (staying in the same state is an option) according to the following distribution: $p(X_{d+1} = \text{Happy} \mid X_d = \text{Angry}) = 0.1$, $p(X_{d+1} = \text{Angry} \mid X_d = \text{Happy}) = 0.1$, $p(X_{d+1} = \text{Angry} \mid X_d = \text{Angry}) = 0.9$, and $p(X_{d+1} = \text{Happy} \mid X_d = \text{Happy}) = 0.9$.
The observation distribution for Harry's Happy state is given by $p(Y_d = \text{smile} \mid X_d = \text{Happy}) = 0.6, p(Y_d = \text{frown} \mid X_d = \text{Happy}) = 0.1, p(Y_d = \text{laugh} \mid X_d = \text{Happy}) = 0.2$, and $p(Y_d = \text{yell} \mid X_d = \text{Happy}) = 0.1$. The observation distribution for Harry's Angry state is $p(Y_d = \text{smile} \mid X_d = \text{Angry}) = 0.1, p(Y_d = \text{frown} \mid X_d = \text{Angry}) = 0.6, p(Y_d = \text{laugh} \mid X_d = \text{Angry}) = 0.1$, and $p(Y_d = \text{yell} \mid X_d = \text{Angry}) = 0.2$.

All of this is summarized in the following figure:



Be sure to show all of your work for the below questions. Note, the goal of this question is to get you to start thinking deeply about probabilistic inference. Thus, although you could look at Chapter 17 for an overview of HMMs, try to solve this question based on first principles (also: no programming needed!).

(a) What is $p(X_2 = \text{Happy})$?

Solution: Given that on day 1 Harry is in the Happy state, on day 2 we have

$$p(X_2 = \text{Happy}) = \sum_{X_1} p(X_1)p(X_2 = \text{Happy}|X_1) = 0.9$$

(b) What is $p(Y_2 = \text{frown})$?

Solution: On day 2 we have $p(X_2 = \text{Happy}) = 0.9$ and $p(X_2 = \text{Angry}) = 0.1$. Since Y_2 is only dependent on X_2, we have

$$\begin{aligned} p(Y_2 = \text{frown}) &= \sum_{X_2} p(X_2)p(Y_2 = \text{frown}|X_2) \\ &= p(X_2 = \text{Happy})p(Y_2 = \text{frown}|X_2 = \text{Happy}) \\ &\quad + p(X_2 = \text{Angry})p(Y_2 = \text{frown}|X_2 = \text{Angry}) \\ &= 0.9 \cdot 0.1 + 0.1 \cdot 0.6 = 0.15 \end{aligned}$$

(c) What is $p(X_2 = \text{Happy} \mid Y_2 = \text{frown})$?

Solution: Using Bayesian rule, we have

$$\begin{aligned} p(X_2 = \text{Happy}|Y_2 = \text{frown}) &= \frac{p(X_2 = \text{Happy})p(Y_2 = \text{frown}|X_2 = \text{Happy})}{p(Y_2 = \text{frown})} \\ &= \frac{0.9 \cdot 0.1}{0.15} = 0.6 \end{aligned}$$

(d) What is $p(Y_{80} = \text{yell})$?

Solution: After a long time, the distribution of hidden states of a HMM converges to its stable distribution. In this case, let $\mathbf{X} = [p(X = \text{Happy}), p(X = \text{Angry})]^\top$, then the stable distribution is given by

$$\begin{bmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{bmatrix} \mathbf{X} = \mathbf{X}$$

$$\mathbf{X} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

Therefore, we have $p(Y_{80}) = \text{yell} = 0.5 \cdot 0.1 + 0.5 \cdot 0.2 = 0.15$.

Alternative approaches:

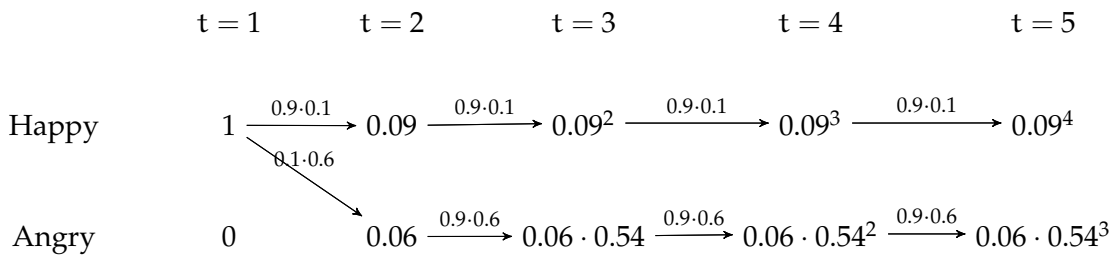
- When considering only the state transition, the two states are symmetric (having the same probability to transit to the other), so it directly follows that they should have the same probability in the stable distribution.
- For a more rigorous approach, one can diagonalize the transition matrix into $\Lambda D \Lambda^{-1}$, then the state distribution on day 80 is given by $\Lambda D^{79} \Lambda^{-1} \mathbf{X}_1$.

(e) Assume that $Y_1 = Y_2 = Y_3 = Y_4 = Y_5 = \text{frown}$. What is the most likely sequence of the states? That is, compute the MAP assignment $\arg\max_{x_1,\ldots,x_5} p(X_1 = x_1, \ldots, X_5 = x_5 \mid Y_1 = Y_2 = Y_3 = Y_4 = Y_5 = \text{frown})$.

Solution: We use the Viterbi algorithm to determine the max likely sequence of the states. At each time and each state, we pick the path that has the maximum probability to end at that state, which can be given by

$$\begin{aligned} p(X_1, \ldots, X_t|Y_1, \ldots, Y_t) &= p(X_1, \ldots, X_{t-1}|Y_1, \ldots, Y_{t-1}) \frac{p(X_t|X_{t-1})p(Y_t|X_t)}{p(Y_t)} \\ &\propto p(X_1, \ldots, X_{t-1}|Y_1, \ldots, Y_{t-1})p(X_t|X_{t-1})p(Y_t|X_t) \end{aligned}$$

In the end we pick the path with the maximum probability. The diagram of Viterbi is shown below.



The path ending in $X_5 = \text{Angry}$ gives the maximum probability, therefore the most likely assignment is (Happy, Angry, Angry, Angry, Angry).

6 Naive Bayes

In this problem you will show that naive Bayes corresponds to a linear classifier. Consider using a naive Bayes algorithm for binary prediction (two classes), where the features $x_1, \ldots, x_k$ are also binary valued. Let $\theta_c = \Pr(Y = c)$ and $\theta_{ci} = \Pr(X_i = 1 | Y = c)$ for $c \in \{0, 1\}$. It will be helpful to use the following form for the joint distribution:

$$\Pr(Y = 1, x_1, \ldots, x_k; \vec{\theta}) = \theta_1 \prod_{i=1}^{k} \theta_{1i}^{x_i} (1 - \theta_{1i})^{1 - x_i}$$

$$\Pr(Y = 0, x_1, \ldots, x_k; \vec{\theta}) = \theta_0 \prod_{i=1}^{k} \theta_{0i}^{x_i} (1 - \theta_{0i})^{1 - x_i}$$

For a naive Bayes model given by parameters $\vec{\theta}$, demonstrate a weight vector $\mathbf{w}$ and offset b such that for any new example $\mathbf{x}$,

$$\arg\max_y \Pr(y | \mathbf{x}; \vec{\theta}) = \arg\max_y y(\mathbf{w} \cdot \mathbf{x} + b)$$

where $\vec{\theta}$ refers to all parameters, including θ_c and θ_{ci}.

Hint: Use Bayes' rule to obtain the posterior, and then take its logarithm (noticing that this is a monotonic transformation which does not change the argmax).

Thus, if one had a sufficient amount of data, one would prefer to directly learn a linear model using logistic regression or a SVM rather than using naive Bayes, since the former consider a strictly larger hypothesis class than the latter. With limited numbers of training points (or settings where some features may be missing) naive Bayes may be preferable.

Solution: In order to calculate $\arg\max_y \Pr(y | \mathbf{x}; \vec{\theta})$, we take the log of the ratio of the condi-

tional probabilities. Using the Bayes' rule $\Pr(y|\mathbf{x};\vec{\theta}) = \frac{\Pr(y,\mathbf{x};\vec{\theta})}{\Pr(\mathbf{x};\vec{\theta})}$, we can define α as

$$\begin{aligned}\alpha &= \log\left(\frac{\Pr(Y=1|\mathbf{x};\vec{\theta})}{\Pr(Y=0|\mathbf{x};\vec{\theta})}\right)\\ &= \log\left(\frac{\Pr(Y=1,\mathbf{x};\vec{\theta})}{\Pr(Y=0,\mathbf{x};\vec{\theta})}\right)\end{aligned}$$

Then the sign of α will determine $\arg\max_y \Pr(y|\mathbf{x};\vec{\theta})$, i.e. $\arg\max_y \Pr(y|\mathbf{x};\vec{\theta}) = 1$ if and only if $\alpha > 0$. In other words, $\arg\max_y \Pr(y|\mathbf{x};\vec{\theta}) = \arg\max_y \alpha y$.

Now we try to rewrite α in the form of $(\mathbf{w}\cdot\mathbf{x}+b)$:

$$\begin{aligned}\alpha &= \log\left(\frac{\Pr(Y=1,\mathbf{x};\vec{\theta})}{\Pr(Y=0,\mathbf{x};\vec{\theta})}\right)\\ &= \log\Pr(Y=1,\mathbf{x};\vec{\theta}) - \log\Pr(Y=0,\mathbf{x};\vec{\theta})\\ &= \left(\log\theta_1 + \sum_{i=1}^{k} x_i\log\theta_{1i} + \sum_{i=1}^{k}(1-x_i)\log(1-\theta_{1i})\right)\\ &\quad - \left(\log\theta_0 + \sum_{i=1}^{k} x_i\log\theta_{0i} + \sum_{i=1}^{k}(1-x_i)\log(1-\theta_{0i})\right)\\ &= \sum_{i=1}^{k} x_i\log\frac{\theta_{1i}(1-\theta_{0i})}{\theta_{0i}(1-\theta_{1i})} + \left(\log\frac{\theta_1}{\theta_0} + \sum_{i=1}^{k}\log\frac{1-\theta_{1i}}{1-\theta_{0i}}\right)\end{aligned}$$

Therefore, by setting $\mathbf{w}$ and b as follows, we can have $\arg\max_y \Pr(y|\mathbf{x};\vec{\theta}) = \arg\max_y y(\mathbf{w}\cdot\mathbf{x}+b)$.

$$\begin{aligned}w_i &= \log\frac{\theta_{1i}(1-\theta_{0i})}{\theta_{0i}(1-\theta_{1i})}\\ b &= \log\frac{\theta_1}{\theta_0} + \sum_{i=1}^{k}\log\frac{1-\theta_{1i}}{1-\theta_{0i}}\end{aligned}$$