CS 337 (Spring 2019): End-semester Examination

Instructor: Shivaram Kalyanakrishnan

5.30 p.m. – 8.30 p.m., April 28, 2019, 101/103/105 New CSE Building

Total marks: 40

Note. Provide brief justifications and/or calculations along with each answer to illustrate how you arrived at the answer.

Question 1. Recall that the "L^0 norm" of a vector V, written $\|V\|_0$, denotes the number of non-zero elements of V. For example, $\|(-5, 0, 0.02)\|_0 = 2$, $\|(1, 1, -1, 0, 1)\|_0 = 4$, and $\|(0, 0, 0)\|_0 = 0$. This question considers the use of this norm for regularisation in linear regression. Consider the following data set with input variables $x_1, x_2 \in \mathbb{R}$ and label $y \in \mathbb{R}$. There are 3 data points.

r	x_1^r	x_2^r	y^r
1	-1	2	3
2	0	1	3
3	1	0	1

For $\mathbf{w} = (w_1, w_2) \in \mathbb{R}^2$, define a loss

$$L(\mathbf{w}) = \left(\sum_{r=1}^{3} (y^r - w_1 x_1^r - w_2 x_2^r)^2\right) + \lambda \|\mathbf{w}\|_0,$$

where $\lambda \geq 0$ is the regularisation coefficient.

1a. Plot $\min_{\mathbf{w} \in \mathbb{R}^2} L(\mathbf{w})$ as a function of λ for $\lambda \in [0, 20]$. [8 marks]

1b. Is the "L^0 norm" commonly used in practice for regularisation? Why or why not? [1 mark]

Question 2. This question is about the use of kernels in machine learning.

2a. In 2-dimensional space $\mathbb{R}^2$, the function $K : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ is a *kernel* if there exists a feature map $F : \mathbb{R}^2 \to \mathbb{R}^d$ for some dimension $d \geq 1$ such that for $\mathbf{x}, \mathbf{z} \in \mathbb{R}^2$, $K(\mathbf{x}, \mathbf{z}) = F(\mathbf{x}) \cdot F(\mathbf{z})$. For each of the following functions, answer whether it is a kernel (in 2-dimensional space), and prove that your answer is correct.

2a(i). $K_1(\mathbf{x}, \mathbf{z}) = 1 + (\mathbf{x} \cdot \mathbf{z})^3$. [2 marks]

2a(ii). $K_2(\mathbf{x}, \mathbf{z}) = (\mathbf{x} + \mathbf{z}) \cdot (\mathbf{x} - \mathbf{z})$. [2 marks]

2b. What is the purpose of using kernels in conjunction with SVMs? Describe one strength and one weakness of kernelised SVMs when compared to neural networks. [2 marks]

Question 3. A data set D contains $n \geq 1$ labelled points, each point being 2-dimensional, real-valued, and its label also real-valued. In other words, $D = \{(\mathbf{x}^1, y^1), (\mathbf{x}^2, y^2), \dots, (\mathbf{x}^n, y^n)\}$, where for $r \in \{1, 2, \dots, n\}$, $\mathbf{x}^r \in \mathbb{R}^2$ and $y^r \in \mathbb{R}$.

If a model produces prediction y^r_{pred} for each data point r, the loss associated with the model is taken to be $\sum_{r=1}^{n}(y^r - y^r_{\text{pred}})^2$. Consider the family of models parameterised by $(\tau_1, \tau_2, \tau_3, \pi_1, \pi_2, \pi_3, \pi_4) \in \mathbb{R}^7$, which make a prediction $y^q_{\text{pred}} \in \mathbb{R}$ for a query point $\mathbf{x}^q \in \mathbb{R}^2$ as follows.

```
If $x^q_1 < \tau_1$:
    If $x^q_2 < \tau_2$:
        $y^q_{\text{pred}} \leftarrow \pi_1$.
    Else
        $y^q_{\text{pred}} \leftarrow \pi_2$.
Else:
    If $x^q_2 < \tau_3$:
        $y^q_{\text{pred}} \leftarrow \pi_3$.
    Else
        $y^q_{\text{pred}} \leftarrow \pi_4$.

Return $y^q_{\text{pred}}$.
```

Describe a procedure to take the data set D as input and compute an *optimal* parameter vector $(\tau_1^\star, \tau_2^\star, \tau_3^\star, \pi_1^\star, \pi_2^\star, \pi_3^\star, \pi_4^\star)$ in the sense that it gives a model with minimum loss on D.

You can use both descriptive statements in English and snippets of pseudocode. If convenient, treat D as an n-sized array of objects (x, y), where x is itself a 2-dimensional array. You can assume access to standard primitives such as testing for membership and sorting. To avoid worries related to tie-breaking, assume no two x_1- or x_2-values are identical in the data set.

Your procedure will primarily be assessed for correctness. However, to obtain full marks, it must run in $O(n^2)$ time. Be sure to provide a sketch arguing for correctness and the claimed time complexity. [7 marks]

Question 4. Consider an MDP (S, A, T, R, γ), with notations being as usual and discount factor $\gamma \in [0, 1)$. Recall that the Value Iteration algorithm produces a sequence $V^0, V^1, V^2, \dots$, each element being a mapping from S to $\mathbb{R}$, which converges to the optimal value function $V^\star$.

4a. For $t = 0, 1, \dots$, write down how V^{t+1} is obtained from V^t. [2 marks]

4b. Assume that there is a scalar $R_{max} > 0$ such that each individual reward obtained from R lies in $[0, R_{\max}]$. Also assume that Value Iteration is initialised with the zero vector: that is, $V^0 = \mathbf{0}$. Show that for $t = 0, 1, \dots$ and for $s \in S$:

$$V^\star(s) - V^t(s) \leq \frac{\gamma^t R_{\max}}{1 - \gamma}. \text{ [5 marks]}$$

Question 5. For a particular search instance, h_1 and h_2 are both *consistent* heuristics, and additionally, for all nodes n, $h_1(n) \geq h_2(n)$. Let $g(n)$ denote the path cost of node n. Assume that there exists an optimal path to goal of finite length. In order to avoid reasoning about ties, assume that the search instance and the heuristics are such that for every pair of distinct nodes n and n', $g(n) + h_1(n) \neq g(n') + h_1(n')$ and $g(n) + h_2(n) \neq g(n') + h_2(n')$.

Let N_1 be the number of expanded nodes if A* search is run with h_1 as the heuristic, and N_2 be the number of expanded nodes if it is run with h_2 as the heuristic. Is it guaranteed that N_1 will not exceed N_2? Or is it guaranteed that N_2 will not exceed N_1? Or can the order between N_1 and N_2 vary depending on the search instance? Justify your answer with a proof. [4 marks]

Question 6. Consider the Bayes Net $X_1 \rightarrow X_2 \rightarrow X_3$, wherein each variable takes Boolean values (without and with negation in corresponding lower case). The conditional probability distributions are as follows; probabilities for negated values are implicit. Assume $a, b, c \in (0, 1)$.

$$\boxed{\mathbb{P}\{x_1\} = a;\ \mathbb{P}\{x_2|x_1\} = \mathbb{P}\{x_3|x_2\} = b;\ \mathbb{P}\{x_2|\neg x_1\} = \mathbb{P}\{x_3|\neg x_2\} = c.}$$

6a. Express the following probabilities in terms of a, b, and c. [2 marks]

- $\mathbb{P}\{x_2|x_1, x_3\}$ (denoted α_1 for use in 6b).
- $\mathbb{P}\{x_2|x_1, \neg x_3\}$ (denoted α_2 for use in 6b).
- $\mathbb{P}\{x_3|x_1, x_2\}$ (denoted α_3 for use in 6b).
- $\mathbb{P}\{x_3|x_1, \neg x_2\}$ (denoted α_4 for use in 6b).

6b. Consider the use of Gibbs Sampling to draw samples from the conditional probability distribution of X_2, X_3 given $X_1 = x_1$. Assume that the Gibbs Sampling process achieves steady state probability γ_1 of being in state (x_1, x_2, x_3), γ_2 of being in state $(x_1, x_2, \neg x_3)$, and γ_3 of being in state $(x_1, \neg x_2, x_3)$. Fill in the blanks below with arithmetic functions of (some or all of) α_1, α_2, α_3, and α_4 such that the resulting recurrence reflects the dynamics of the Gibbs Sampling process.

$$\gamma_1 = (\_\_\_\_\_\_\_\_ \times \gamma_1) + (\_\_\_\_\_\_\_\_ \times \gamma_2) + (\_\_\_\_\_\_\_\_ \times \gamma_3).$$ [2 marks]

6c. Express $\mathbb{P}\{x_2, x_3|x_1\}$, $\mathbb{P}\{x_2, \neg x_3|x_1\}$, and $\mathbb{P}\{\neg x_2, x_3|x_1\}$ in terms of a, b, and c. Does taking $\gamma_1 = \mathbb{P}\{x_2, x_3|x_1\}$, $\gamma_2 = \mathbb{P}\{x_2, \neg x_3|x_1\}$, $\gamma_3 = \mathbb{P}\{\neg x_2, x_3|x_1\}$ satisfy the recurrence you have written in 6b? [3 marks]

Solutions

1a. We partition $\mathbb{R}^2$ into three sets:

- $S_2 = \{(w_1, w_2) \in \mathbb{R}^2 : w_1 \neq 0, w_2 \neq 0\}$.
- $S_1 = \{(w_1, w_2) \in \mathbb{R}^2 : (w_1 \neq 0, w_2 = 0) \vee (w_1 = 0, w_2 \neq 0)\}$.
- $S_0 = \{(w_1, w_2) \in \mathbb{R}^2 : w_1 = 0, w_2 = 0\}$.

The idea behind the partitioning is that (1) the elements $\mathbf{w}$ of each set have the same value of $\|\mathbf{w}\|_0$ and (2) within each set, the minimiser and mimumum of $L(\mathbf{w})$ are easy to find. We find each minimum in turn.

$$\underset{(w_1,w_2)\in S_2}{\text{argmin}} \; L((w_1, w_2)) = \underset{(w_1,w_2)\in S_2}{\text{argmin}} \; ((3 + w_1 - 2w_2)^2 + (3 - w_2)^2 + (1 - w_1)^2)) = \left(\frac{4}{3}, \frac{7}{3}\right);$$

$$\min_{(w_1,w_2)\in S_2} L((w_1, w_2)) = \frac{2}{3} + 2\lambda.$$

The minimum from S_1 is determined by considering two cases:

$$\begin{aligned}
&\min_{(w_1,w_2)\in S_1} L((w_1, w_2) \\
&= \min\{\min_{(w_1\neq 0,w_2=0)} L((w_1, w_2)), \min_{(w_1=0,w_2\neq 0)} L((w_1, w_2))\} \\
&= \min\{\min_{(w_1\neq 0,w_2=0)} ((3 + w_1)^2 + (1 - w_1)^2 + 9 + \lambda), \min_{(w_1=0,w_2\neq 0)} ((3 - 2w_2)^2 + (3 - w_2)^2 + 1 + \lambda)\} \\
&= \min\{17 + \lambda, \frac{14}{5} + \lambda\} = \frac{14}{5} + \lambda.
\end{aligned}$$

S_0 only has a single element, $(0, 0)$, and so

$$\min_{(w_1,w_2)\in S_0} L((w_1, w_2)) = 3^2 + 3^2 + 1^2 + (0)\lambda = 19.$$

Now, we obtain

$$\min_{\mathbf{w}\in\mathbb{R}^2} L(\mathbf{w}) = \min\{\min_{\mathbf{w}\in S_2} L(\mathbf{w}), \min_{\mathbf{w}\in S_1} L(\mathbf{w}), \min_{\mathbf{w}\in S_0} L(\mathbf{w})\} = \min\left\{\frac{2}{3} + 2\lambda, \frac{14}{5} + \lambda, 19\right\}.$$

The function is a continuous, non-decreasing, piecewise linear function of λ, given by

$$\min_{\mathbf{w}\in\mathbb{R}^2} L(\mathbf{w}) = \begin{cases} \frac{2}{3} + 2\lambda, 0 \leq \lambda \leq \frac{32}{15}, \\ \frac{14}{5} + \lambda, \frac{32}{15} < \lambda \leq 16, \\ 19, \lambda > 16. \end{cases}$$

1b. The "L^0 norm" is *not* used commonly in practice for regularisation because there are no known techniques to solve the resulting optimisation problem in a computationally-efficient manner. In 1a, the problem was 2-dimensional, and so we could enumerate the few possible configurations of weights corresponding to each possible value of $\|\mathbf{w}\|_0$. However, the number of configurations is exponential in the number of dimensions. Our approach would not be feasible in higher dimensions.

2a(i). For $\mathbf{x} \in \mathbb{R}$, take $F(\mathbf{x}) = (1, (x_1)^3, (x_2)^3, \sqrt{3}(x_1)^2 x_2, \sqrt{3}x_1(x_2)^2)$. Then

$$\begin{aligned} F(\mathbf{x}) \cdot F(\mathbf{z}) &= 1 + (x_1)^3(z_1)^3 + (x_2)^3(z_2)^3 + 3(x_1)^2 x_2 (z_1)^2 z_2 + 3x_1(x_2)^2 z_1 (z_2)^2 \\ &= 1 + (x_1 z_1 + x_2 z_2)^3 = 1 + (\mathbf{x} \cdot \mathbf{z})^3 = K_1(\mathbf{x}, \mathbf{z}), \end{aligned}$$

implying that K_1 is a kernel function.

2a(ii). Assume that for some feature map F, indeed $K_2(\mathbf{x}, \mathbf{z}) = F(\mathbf{x}) \cdot F(\mathbf{z})$. Then, we see that $K_2(\mathbf{z}, \mathbf{x}) = F(\mathbf{z}) \cdot F(\mathbf{x})$ must be equal to $K_2(\mathbf{x}, \mathbf{z})$: in other words, K_2 must be symmetric with respect to its arguments. Clearly it is not, and so cannot be a kernel function.

2b. The basic SVM formulation, say for binary classification, is designed to find a *linear* separator between two sets of points. However, in practice, the points might need a non-linear separating boundary. Kernels implicitly transform the input points to a higher-dimensional space wherein they are hoped to be "more separable".

Whether used with or without kernels, SVM training amounts to solving a convex optimisation problem, which guarantees an optimal solution. This aspect is in contrast with neural network training, which optimises a non-convex function, and can get stuck at local optima. However, neural networks have the advantage of inferring higher-order representations from the data (in their many hidden layers). Traditional usage of SVMs with kernels usually implies applying the same non-linear operations (or a few from a small set) regardless of the data set, which limits the practical efficacy of the method.

3. From the template of the model, it is apparent that the points are partitioned into four clusters based on their x_1 and x_2 coordinates, and a single prediction is associated with each cluster. The easy portion to first get out of the way is to observe that for any fixed clustering (determined by τ_1, τ_2, and τ_3), the prediction made for each cluster must be its mean y value (which minimises the SSE for that cluster). That settles $\pi_1^\star$ $\pi_2^\star$, $\pi_3^\star$, and $\pi_4^\star$, provided we find $\tau_1^\star$, $\tau_2^\star$, and $\tau_3^\star$. We find these optimal thresholds by examining all the relevant (τ_1, τ_2) and (τ_1, τ_3) pairs.

There are $n-1$ possible ways to divide the points based on x_1 into sets of size $m \in \{1, 2, \ldots, n-1\}$ and $n - m$; it suffices to consider $n - 1$ values for τ_1. If τ_1 results in an m:$(n - m)$ split, similarly it suffices to consider $m - 1$ values for τ_2 and $n - m - 1$ values for τ_3. Further, τ_2 and τ_3 *independently* determine the loss of two separate sets of points, which means $\theta(n^2)$ *evaluations* suffice to identify $(\tau_1^\star, \tau_2^\star, \tau_3^\star)$. We have implicitly assumed that for given (τ_1, τ_2, τ_3), there is no additional overhead to create/store/represent the four resulting clusters. This assumption is justified if we initially sort the points on x_1 and x_2—which should take no more than $\theta(n \log n)$ time.

A naïve way to implement our solution would be to calculate the SSE of each cluster for each of the $\theta(n^2)$ settings of (τ_1, τ_2) (and (τ_1, τ_3)), but this approach would result in an overall complexity of $\theta(n^3)$. To gain efficiency, consider that the $m - 1$ clusterings obtained by varying τ_2 in sequence (while keeping τ_1 fixed) will progress from a 1:(m - 1) split, incrementally, all the way to an $m - 1$:1 split. Conveniently, the SSE of an $(a - 1)$:$(b + 1)$ split can be obtained in only $\theta(1)$ time from the SSE of an a:b split, as long as we store the number of points a, sums of the y-coordinates sum_y and the sums of the squares of the y-coordinates $sumSquared_y$ of each cluster. Clearly these quantities can be updated in $\theta(1)$ time as points are added or removed. The SSE is given by $sumSquared_y - a \cdot (sum_y)^2$ and the mean is given by sum_y/a.

4a. For $t = 0, 1, \ldots$ and $s \in S$:

$$V^{t+1}(s) \leftarrow \max_{a \in A} \sum_{s' \in S} T(s, a, s')\{R(s, a, s') + \gamma V^t(s')\}.$$

4b. We prove the result by induction on t. For $t = 0$, we have to show that for $s \in S$,

$$V^\star(s) \leq \frac{R_{\max}}{1 - \gamma},$$

which is evident since $V^\star(s)$ is the expected infinite discounted reward obtained by a policy, in this case an optimal policy. Even if each reward is maximum, $V^\star(s)$ can at most be

$$R_{\max} + \gamma R_{\max} + \gamma^2 R_{\max} + \cdots = \frac{R_{\max}}{1 - \gamma}.$$

Assume the result is true for some $t \geq 0$. Take $\pi^\star$ to be any optimal policy. We have, for $s \in S$:

$$\begin{aligned} V^{t+1}(s) &= \max_{a \in A} \sum_{s' \in S} T(s, a, s')\{R(s, a, s') + \gamma V^t(s')\} \\ &\geq \sum_{s' \in S} T(s, \pi^\star(s), s')\{R(s, \pi^\star(s), s') + \gamma V^t(s')\} \\ &\geq \sum_{s' \in S} T(s, \pi^\star(s), s') \left\{ R(s, \pi^\star(s), s') + \gamma \left(V^\star(s') - \frac{\gamma^t R_{\max}}{1 - \gamma} \right) \right\}, \end{aligned}$$

wherein the last step applies the induction hypothesis and also the fact that the transition probabilities are non-negative. By expanding out, we get

$$\begin{aligned} V^{t+1}(s) &\geq \sum_{s' \in S} T(s, \pi^\star(s), s') \left\{ R(s, \pi^\star(s), s') + \gamma (V^\star(s') \right\} - \sum_{s' \in S} T(s, \pi^\star(s), s') \left\{ \gamma \left(\frac{\gamma^t R_{\max}}{1 - \gamma} \right) \right\} \\ &= V^\star(s) - \frac{\gamma^{t+1} R_{\max}}{1 - \gamma}, \end{aligned}$$

obtained by invoking Bellman's Equations for $\pi^\star$ and equating the sum of transition probabilities from $(s, \pi^\star(s))$ to 1.

A second, direct approach would be to split

$$V^\star(s) = \mathbb{E}_{\pi^\star}\{r^0 + \gamma r^1 + \gamma^2 r^2 + \ldots | s^0 = s\}$$

into a sum of

$$T_1 = \mathbb{E}_{\pi^\star}\{r^0 + \gamma r^1 + \gamma^2 r^2 + \cdots + \gamma^{t-1} r^{t-1} | s^0 = s\}$$

and

$$T_2 = \mathbb{E}_{\pi^\star}\{\gamma^t r^t + \gamma^{t+1} r^{t+1} + \ldots | s^0 = s\}.$$

It so happens that $V^t(s)$ is the maximum expected discounted t-step reward that can be possibly obtained; in general one would have to follow a *non-stationary* (time-dependent) policy in order to achieve it. T_1 is the expected discounted t-step reward obtained by following $\pi^\star$, and so cannot exceed $V^t(s)$. Since each individual reward is upper-bounded by $R_{\max}$, we see that T_2 is at most $\frac{\gamma^t R_{\max}}{1-\gamma}$. Hence, $V^\star(s) \leq V^t(s) + \frac{\gamma^t R_{\max}}{1-\gamma}$.

5. Let R1 denote the run of A* using h_1 as the heuristic, and let R2 be the run of A* using h_2 as the heuristic. Since both heuristics are consistent, both will find an optimal-cost goal node G. Since we have assumed that no two nodes have the same $g + h_1$ (or $g + h_2$) value, and since goal nodes have a zero heuristic value, we infer that G must be the unique optimal goal node. Let its cost be $C^\star$.

We also know that $A^\star$ expands nodes in non-decreasing order of $g+h$, which implies R1 expands every node n such that $g(n)+h_1(n) \leq C^\star$ and R2 expands every node n such that $g(n)+h_2(n) \leq C^\star$. Since for every node n, $h_1(n) \geq h_2(n)$, it follows that $g(n) + h_1(n) \leq C^\star \implies g(n) + h_2(n) \leq C^\star$. In other words, every node expanded by R1 is also expanded by R2 (the converse need not be true). We have thus proven $N_1 \leq N_2$.

6a.

$$\alpha_1 = \mathbb{P}\{x_2|x_1, x_3\} = \frac{\mathbb{P}\{x_1, x_2, x_3\}}{\mathbb{P}\{x_1, x_2, x_3\} + \mathbb{P}\{x_1, \neg x_2, x_3\}} = \frac{abb}{abb + a(1-b)c} = \frac{b^2}{b^2 + (1-b)c}.$$

$$\alpha_2 = \mathbb{P}\{x_2|x_1, \neg x_3\} = \frac{\mathbb{P}\{x_1, x_2, \neg x_3\}}{\mathbb{P}\{x_1, x_2, \neg x_3\} + \mathbb{P}\{x_1, \neg x_2, \neg x_3\}} = \frac{ab(1-b)}{ab(1-b) + a(1-b)(1-c)} = \frac{b}{b+1-c}.$$

$$\alpha_3 = \mathbb{P}\{x_3|x_1, x_2\} = \frac{\mathbb{P}\{x_1, x_2, x_3\}}{\mathbb{P}\{x_1, x_2, x_3\} + \mathbb{P}\{x_1, x_2, \neg x_3\}} = \frac{abb}{abb + ab(1-b)} = b.$$

$$\alpha_4 = \mathbb{P}\{x_3|x_1, \neg x_2\} = \frac{\mathbb{P}\{x_1, \neg x_2, x_3\}}{\mathbb{P}\{x_1, \neg x_2, x_3\} + \mathbb{P}\{x_1, \neg x_2, \neg x_3\}} = \frac{a(1-b)c}{a(1-b)c + a(1-b)(1-c)} = c.$$

6b. Recall that to go from one state to another, Gibbs Sampling selects a non-given variable uniformly at random and sets it to a value sampled from its distribution given all the other variables in the current state. Therefore, in our example, the probability of going

- from (x_1, x_2, x_3) to (x_1, x_2, x_3) is $\frac{1}{2}\alpha_1 + \frac{1}{2}\alpha_3$;
- from $(x_1, \neg x_2, x_3)$ to (x_1, x_2, x_3) is $\frac{1}{2}\alpha_1$;
- from $(x_1, x_2, \neg x_3)$ to (x_1, x_2, x_3) is $\frac{1}{2}\alpha_3$; and
- from $(x_1, \neg x_2, \neg x_3)$ to (x_1, x_2, x_3) is 0.

Consequently the steady state probabilities satisfy

$$\gamma_1 = \frac{\alpha_1 + \alpha_3}{2}\gamma_1 + \frac{\alpha_3}{2}\gamma_2 + \frac{\alpha_1}{2}\gamma_3.$$

6c.

$$\mathbb{P}\{x_2, x_3|x_1\} = \frac{\mathbb{P}\{x_1, x_2, x_3\}}{\mathbb{P}\{x_1\}} = \frac{abb}{a} = b^2.$$

$$\mathbb{P}\{x_2, \neg x_3|x_1\} = \frac{\mathbb{P}\{x_1, x_2, \neg x_3\}}{\mathbb{P}\{x_1\}} = \frac{ab(1-b)}{a} = b(1-b).$$

$$\mathbb{P}\{\neg x_2, x_3|x_1\} = \frac{\mathbb{P}\{x_1, \neg x_2, x_3\}}{\mathbb{P}\{x_1\}} = \frac{a(1-b)c}{a} = (1-b)c.$$

It can be verified that taking $\gamma_1 = \mathbb{P}\{x_2, x_3|x_1\}$, $\gamma_2 = \mathbb{P}\{x_2, \neg x_3|x_1\}$, $\gamma_3 = \mathbb{P}\{\neg x_2, x_3|x_1\}$ indeed satisfies the recurrence in 6b.

In class we did not undertake the proof of the consistency of Gibbs Sampling, which proceeds by analysing a Markov Chain. In this example, too, we have not furnished a full proof—we have only verified that the true probabilities of a subset of samples satisfy the steady state equation of the Gibbs Sampling process. Nonetheless, the example is illustrative.