CS 181 Artificial Intelligence (Fall 2019), Final Exam

Instructions

- Time: 14:00 – 15:40 (100 minutes)
- This exam is closed-book, but you may bring one A4-size cheat sheet. Put all the study materials and electronic devices into your bag and put your bag in the front, back, or sides of the classroom.
- Write all your answers **on the answer sheets** (which are attached after the problem sheets). We will only grade the answer sheets and will not look at the problem sheets.
- Write your email address on the top of **every** answer sheet.
- Three blank pieces of paper are attached after the problem sheets, which you can use as scratch paper. Raise your hand if you need more paper.

1 Multiple choices (10 pts)

Each question has one or more correct answers. Select all the correct answers. For each question, you get 1 point if your select all the correct answers and nothing else, 0 point if you select one or more wrong answers, and 0.5 point if you select a non-empty proper subset of the correct answers.

1. Which of the following statements about the hidden Markov model (HMM) is/are correct?
   A. The time complexity of the Viterbi algorithm is linear in the number of hidden states.
   B. If a dynamic Bayes net is discrete, there must exist a hidden Markov model that can represent it.
   C. If the hidden state at time t is given, both the evidence at time t and the hidden state at time $t+1$ are independent of the hidden state at time $t-1$.
   D. Without the resampling step, particle filtering performs similarly to likelihood weighting.
2. Which of the following statements about the Markov decision process (MDP) is/are correct? S and A denote the sizes of the state space and action space respectively.
   A. Once a policy of an MDP is decided, the MDP performs like a Markov chain according to the policy.
   B. In value iteration, the vector V_k will converge as long as the discount $\gamma < 1$.
   C. Policy iteration takes $O(S^2A)$ per iteration.
   D. Policy iteration is guaranteed to be faster than value iteration.
   E. None of the above.

3. Which of the following statements about reinforcement learning is/are correct?
   A. Reinforcement learning is offline planning because the agent has no prior knowledge of rewards or transitions in the world.
   B. Model-free learning attempts to estimate the values or Q-values of states directly, without constructing a model of the rewards and transitions in the MDP.
   C. Direct evaluation follows a policy π and makes use of the information about transitions between states. It is slow because it requires enough samples in order to converge to the true values of states under π.
   D. In temporal difference learning, older samples are given linearly less weights.
   E. None of the above.

4. Which of the following statements about Q-learning is/are correct?
   A. Q-learning is a passive reinforcement learning method.
   B. The ϵ-greedy strategy often produces a lower regret for a decaying ϵ than a fixed one.
   C. Q-learning can learn an optimal policy only when the regret is small enough.
   D. Compared with Q-learning, approximate Q-learning that describes each state with a vector of features can be helpful when the number of states is huge.
   E. None of the above.

5. In approximate Q-learning, we use a feature representation $Q(s,a) = \sum_{i=1}^{d} w_i f_i(s,a)$ for weights $w_1, \ldots, w_d$ and feature functions $f_1(s,a), \ldots, f_d(s,a)$. Suppose we experience a transition (s_t, a_t, s_{t+1}) with reward r_t. Which of the following statements is/are correct?
   A. $Q_{\text{sample}} \leftarrow r_t + \gamma \max_{a'} Q(s_t, a')$.
   B. $w_i \leftarrow w_i + \alpha(Q(s_t, a_t) - Q_{\text{sample}}) f_i(s_t, a_t)$.
   C. $w_i \leftarrow w_i + \alpha(Q(s_t, a_t) - Q_{\text{sample}}) w_i$.
   D. Approximate Q-learning is model-free learning.
   E. None of the above.

6. Which of the following statements about regression is/are correct?
   A. Regression is a type of supervised learning.
   B. The least squares method minimizes the sum of errors over the examples.
   C. Overfitting could happen when there are too many training examples but only a few features.
   D. The higher complexity the regression model has, the better it can fit the test data.
   E. None of the above.

7. Which of the following statements about supervised learning is/are correct?
   A. Using a large number of training samples can help avoid overfitting.
   B. For conditional probabilities $P(X|Y)$, Laplace smoothing often performs well when both $|X|$ and $|Y|$ are large.
   C. We should pick the set of hyper-parameters that performs best on the test set.
   D. The better we fit the training set, the better results on the test set we will get.
   E. None of the above.

8. Which of the following statements about unsupervised learning is/are correct?

   A. No matter how the initial points are chosen, the k-means algorithm can always converge.

   B. The EM algorithm can be used to learn any model with hidden variables (missing data).

   C. In M-step of the EM algorithm we compute distributions over hidden variables based on the current parameter values.

   D. When using the EM algorithm to learn a Gaussian mixture model, the M-step computes weighted maximum likelihood estimation (MLE).

   E. None of the above.

9. Which of the following statements about context-free grammars (CFG) is/are correct?

   A. Any arbitrary CFG can be rewritten into the Chomsky normal form (CNF).

   B. A CFG and its corresponding CNF can generate the same set of strings and parse trees.

   C. Probabilistic CFGs tend to assign larger probabilities to smaller parse trees.

   D. Probabilistic CFGs have no ambiguity problem.

   E. None of the above.

10. Which of the following statements about dependency grammars is/are correct?

   A. Dependency grammars focus on constituents.

   B. In first-order graph-based parsing, each connected pair of arcs has a score.

   C. CFG-based parsers are generally faster than dependency parsing.

   D. CFGs are a subclass of dependency grammars.

   E. None of the above.

Solution:

1. BCD
2. ABC
3. B
4. BD
5. D
6. A
7. A
8. ABD
9. AC
10. E

2 Hidden Markov Model (10 pts)

Consider an HMM with hidden states $\{X_t\}$ and evidences $\{E_t\}$. Each X_t takes its value from $\{1, 2\}$ and each E_t takes its value from $\{1, 2, 3\}$.

2.1 Stationary Distribution (2 pts)

Suppose the transition matrix is given by

$$Tran = \begin{bmatrix} 0.7 & 0.3 \\ 0.5 & 0.5 \end{bmatrix}.$$

Note: Recall that the transition matrix represents a family of conditionals $P(X_t|X_{t-1})$. It should be easy for you to figure out whether each row or each column of the matrix represents a conditional distribution.

Compute the stationary distributions $\lim_{t\to\infty} P(X_t = 1)$ and $\lim_{t\to\infty} P(X_t = 2)$.

Solution:

$\lim_{t\to\infty} P(X_t = 1) = 5/8, \quad \lim_{t\to\infty} P(X_t = 2) = 3/8.$

2.2 Most Likely Explanation (4 pts)

Suppose the emission matrix is given by

$$Emi = \begin{bmatrix} 0.8 & 0.3 \\ 0.1 & 0.5 \\ 0.1 & 0.2 \end{bmatrix},$$

and the initial state distribution is given by

$$P(X_0) = \begin{bmatrix} 0.4 \\ 0.6 \end{bmatrix}.$$

Suppose the evidences are observed as $E_1 = 1, E_2 = 2, E_3 = 3$. Compute the most likely explanation $\text{argmax}_{X_{0:3}} P(X_{0:3}|E_{1:3})$ and the likelihood $\max_{X_{0:3}} P(X_{0:3}, E_{1:3})$.

Solution:

$\text{argmax}_{X_{0:3}} P(X_{0:3}|E_{1:3}) = (2, 1, 2, 2), \quad \max_{X_{0:3}} P(X_{0:3}, E_{1:3}) = 0.0036.$

2.3 Backward Algorithm (4 pts)

We have learned the forward algorithm. Here we discuss a similar algorithm called the backward algorithm, which is used to compute $\beta_t(X_t) = P(E_{t+1:n}|X_t)$. We start with $\beta_n(X_n) = 1$ and then recursively compute $\beta_t(X_t)$ for $t = n-1, \cdots, 1$. Derive the recursive formula, i.e., how to compute $\beta_t(X_t)$ from $\beta_{t+1}(X_{t+1})$, Emi and $Tran$.

Solution:

$\beta_t(X_t) = \sum_{X_{t+1}} \beta_{t+1}(X_{t+1}) Emi(E_{t+1}|X_{t+1}) Tran(X_{t+1}|X_t)$

3 MDP + Reinforcement Learning (10 pts)

3.1 Monty Hall's New Game Show (6 pts)

The famous Monty Hall hosts a new game show and invites you to be a player. There are three closed doors $1, 2, 3$. Suppose door i hides i dollar(s) behind it. On each round of the game, the player has the option of opening the door or switching the door:

1. *Open*: Stop playing by opening the door and collecting the dollars that the door hides behind.
2. *Switch*: Switch to one of the other two doors with equal probability and enter the next round. Each switch action costs 1 dollar.

Having taken CS181, you decide to model this problem using an infinite Markov decision process (MDP). The player is initially assigned a door at the very beginning. State s_i denotes the state that the player is in front of door i. Once a player decides to *Open* at s_i, he will collect i dollar(s) and the game is over, transferring the player to the *End* state. We assume $\gamma = 1$ in this game. Answer the following questions.

Note: To get points for each question, you must show the correct calculation process.

3.1.1

In solving this problem, you consider using policy iteration. Your initial policy π_0 is in the table below. Evaluate the policy at each state.

State	s_1	s_2	s_3
$\pi_0(s)$	*Switch*	*Switch*	*Open*
$V^{\pi_0}(s)$	1	1	3

Solution:

Since $\pi_0(s_3) = Open$, $V^{\pi_0}(s_3) = 3$. Then

$$\begin{aligned} V^{\pi_0}(s_1) &= \sum_{s'} T(s_1, \pi(s_1), s')(R(s_1, \pi(s_1), s') + \gamma V^{\pi_0}(s')) \\ &= \frac{1}{2}(-1 + V^{\pi_0}(s_2)) + \frac{1}{2}(-1 + V^{\pi_0}(s_3)). \\ V^{\pi_0}(s_2) &= \frac{1}{2}(-1 + V^{\pi_0}(s_1)) + \frac{1}{2}(-1 + V^{\pi_0}(s_3)) \\ &= -1 + \frac{1}{2}(V^{\pi_0}(s_1) + V^{\pi_0}(s_3)). \end{aligned}$$

Solving this linear system yields $V^{\pi_0}(s_1) = V^{\pi_0}(s_2) = 1$.

3.1.2

Having determined values of V^{π_0}, perform a policy update to find the new policy π_1. The table below shows the old policy π_0. If both *Switch* and *Stop* are viable new actions for a state, then choose *Switch*.

State	s_1	s_2	s_3
$\pi_0(s)$	*Switch*	*Switch*	*Open*
$\pi_1(s)$	*Switch*	*Open*	*Open*

Solution:

$$\begin{aligned}
\pi_1(s_1) &= \text{argmax}\{\textit{Open}: 1,\ \textit{Switch}: \ -1 + \frac{1}{2}(V^{\pi_0}(s_2) + V^{\pi_0}(s_3))\} \\
&= \text{argmax}\{\textit{Open}: 1,\ \textit{Switch}: 1\} \\
&= \{\textit{Switch}\}. \\
\pi_1(s_2) &= \text{argmax}\{\textit{Open}: 2,\ \textit{Switch}: \ -1 + \frac{1}{2}(V^{\pi_0}(s_1) + V^{\pi_0}(s_3))\} \\
&= \text{argmax}\{\textit{Open}: 2,\ \textit{Switch}: 1\} \\
&= \{\textit{Open}\}. \\
\pi_1(s_3) &= \text{argmax}\{\textit{Open}: 3,\ \textit{Switch}: \ -1 + \frac{1}{2}(V^{\pi_0}(s_1) + V^{\pi_0}(s_2))\} \\
&= \text{argmax}\{\textit{Open}: 3,\ \textit{Switch}: 0\} \\
&= \{\textit{Open}\}.
\end{aligned}$$

3.1.3

Is π_1 optimal? Use the following table to explain why or why not. Show your calculation process.

Note: If at some stage there is no need to continue calculating, you can stop filling the rest of the table.

State	s_1	s_2	s_3
$\pi_0(s)$	*Switch*	*Switch*	*Open*
$V^{\pi_0}(s)$	1	1	3
$\pi_1(s)$	*Switch*	*Open*	*Open*
$V^{\pi_1}(s)$	3/2	2	3
$\pi_2(s)$	*Switch*	*Open*	*Open*
$V^{\pi_2}(s)$			
$\pi_3(s)$			

Solution:

Yes.

Since $\pi_1(s_2) = \pi_1(s_3) = \textit{Open}$, $V^{\pi_1}(s_2) = 2$ and $V^{\pi_1}(s_3) = 3$. Then

$$V^{\pi_1}(s_1) = \frac{1}{2}(-1 + V^{\pi_1}(s_2)) + \frac{1}{2}(-1 + V^{\pi_1}(s_3)) = -1 + \frac{1}{2}(2+3) = \frac{3}{2}.$$

Then we perform a policy update to find the new policy π_2.

$$\begin{aligned}
\pi_2(s_1) &= \operatorname{argmax}\{Open\colon 1,\ Switch\colon\ -1+\frac{1}{2}(V^{\pi_1}(s_2)+V^{\pi_1}(s_3))\} \\
&= \operatorname{argmax}\{Open\colon 1,\ Switch\colon\ \frac{3}{2}\} \\
&= \{Switch\}. \\
\pi_2(s_2) &= \operatorname{argmax}\{Open\colon 2,\ Switch\colon\ -1+\frac{1}{2}(V^{\pi_1}(s_1)+V^{\pi_1}(s_3))\} \\
&= \operatorname{argmax}\{Open\colon 2,\ Switch\colon\ \frac{5}{4}\} \\
&= \{Open\}. \\
\pi_2(s_3) &= \operatorname{argmax}\{Open\colon 3,\ Switch\colon\ -1+\frac{1}{2}(V^{\pi_1}(s_1)+V^{\pi_1}(s_2))\} \\
&= \operatorname{argmax}\{Open\colon 3,\ Switch\colon\ \frac{3}{4}\} \\
&= \{Open\}.
\end{aligned}$$

π_2 is the same as π_1, meaning the policy iteration has converged. Since policy iteration converges to the optimal policy, we can be sure that both π_1 and π_2 are optimal.

3.2 Monty Hall's New Game Show Plus (4 pts)

Monty Hall has upgraded his game and now you do not know the underlying MDP. There are still three doors (states) denoted as D_1, D_2 and D_3, and you have two actions to choose from: $\{Switch, Open\}$. This time, you do not know the outcome of each action, i.e., you do not know the transitions nor the rewards. Suppose you choose actions according to some policy π and generate the following sequence of actions and rewards:

t	s_t	a_t	s_{t+1}	r_t
1	D_1	*Switch*	D_2	3
2	D_2	*Switch*	D_3	-2
3	D_3	*Open*	D_3	1
4	D_3	*Open*	D_1	4

Assume a discount factor $\gamma = 0.5$ and a learning rate $\alpha = 0.5$.

Fill the blanks below and you do not need to show your calculation process.

3.2.1

In model-based reinforcement learning, we first estimate the transition function $T(s, a, s')$ and the reward function $R(s, a, s')$. Fill in the estimates of T based on the experience above.

$\hat{T}(D_2,\ Switch,\ D_3) =$ ___1___, $\qquad \hat{T}(D_3,\ Open,\ D_2) =$ ___0___.

3.2.2

Assume that all the Q-values are initialized to 0. What are the Q-values learned by running Q-learning with all the transitions shown above? Note that we update Q-values immediately after we receive a sample.

$Q(D_1,\ Switch) =$ ___3/2___, $\qquad Q(D_3,\ Open) =$ ___21/8___.

Solution:

Recall the update function of Q-learning is:

$$Q(s_t, a_t) \leftarrow (1-\alpha)Q(s_t, a_t) + \alpha(r_t + \gamma \max_{a'} Q(s_{t+1}, a')).$$

Then

$$\begin{aligned}
t = 1\colon Q(D_1, \mathit{Switch}) &= (1-\alpha)Q(D_1, \mathit{Switch}) + \alpha(r_0 + \gamma \max_{a'} Q(S_1, a') \\
&= 0.5 \cdot 0 + 0.5(3 + 0.5 \cdot \max(Q(D_2, \mathit{Switch}), Q(D_2, \mathit{Open}))) \\
&= 0.5 \cdot 0 + 0.5(3 + 0.5 \cdot \max(0, 0)) \\
&= \frac{3}{2}. \\
t = 2\colon Q(D_2, \mathit{Switch}) &= (1-\alpha)Q(D_2, \mathit{Switch}) + \alpha(r_1 + \gamma \max_{a'} Q(S_2, a') \\
&= 0.5 \cdot 0 + 0.5(-2 + 0.5 \cdot \max(Q(D_3, \mathit{Switch}), Q(D_3, \mathit{Open}))) \\
&= 0.5 \cdot 0 + 0.5(-2 + 0.5 \cdot \max(0, 0)) \\
&= -1. \\
t = 3\colon Q(D_3, \mathit{Open}) &= (1-\alpha)Q(D_3, \mathit{Open}) + \alpha(r_2 + \gamma \max_{a'} Q(S_3, a') \\
&= 0.5 \cdot 0 + 0.5(1 + 0.5 \cdot \max(Q(D_3, \mathit{Switch}), Q(D_3, \mathit{Open}))) \\
&= 0.5 \cdot 0 + 0.5(1 + 0.5 \cdot \max(0, 0)) \\
&= \frac{1}{2}. \\
t = 4\colon Q(D_3, \mathit{Open}) &= (1-\alpha)Q(D_3, \mathit{Open}) + \alpha(r_3 + \gamma \max_{a'} Q(S_4, a') \\
&= 0.5 \cdot \frac{1}{2} + 0.5(4 + 0.5 \cdot \max(Q(D_1, \mathit{Switch}), Q(D_1, \mathit{Open}))) \\
&= 0.5 \cdot \frac{1}{2} + 0.5(4 + 0.5 \cdot \max(3/2, 0)) \\
&= \frac{21}{8}.
\end{aligned}$$

4 Supervised and Unsupervised Machine Learning (10 pts)

Mike loves ShanghaiTech so much that he wants to design an 'Email Filter' that only accepts emails that are related to ShanghaiTech. He decides to use **Naive Bayes** as the classification model and use the occurrence (whether the word appears in the email or not) of the following words as features. 'Project', 'Vacation', 'Deadline', 'Relaxing'. We denote the class variable as Y,

$$Y = \begin{cases} 1, & \text{email is related to ShanghaiTech,} \\ 0, & \text{otherwise.} \end{cases}$$

We denote the feature variables as X_w, where $w \in \{$ 'Project', 'Vacation', 'Deadline', 'Relaxing' $\}$ and

$$X_w = \begin{cases} 1, & \text{email contains word } w, \\ 0, & \text{otherwise.} \end{cases}$$

4.1 Supervised Naive Bayes (5 pts)

Answer the following questions about supervised Naive Bayes.

4.1.1 Joint Distribution (1 pt)

Write down the expression of joint distribution $P(Y, X_{\text{Project}}, X_{\text{Vacation}}, X_{\text{Deadline}}, X_{\text{Relaxing}})$.

Solution:

$$P(Y, X_{\text{Project}}, X_{\text{Vacation}}, X_{\text{Deadline}}, X_{\text{Relaxing}}) = P(Y)\Pi_w P(X_w|Y),$$
$$w \in \{ \text{'Project', 'Vacation', 'Deadline', 'Relaxing'} \}.$$

4.1.2 Supervised Learning of Naive Bayes (4 pts)

Mike annotated (assigning a label for each email) four training samples as shown in the following table.

Y	X_{Project}	X_{Vacation}	X_{Deadline}	X_{Relaxing}
1	1	0	1	0
0	1	1	0	0
1	1	1	1	0
0	0	1	1	1

You need to estimate the probability distribution $P(Y)$ and $P(X_w|Y)$, and use it to build a Naive Bayes classifier. Please **fill the following table**, which represents the parameters of a Naive Bayes classifier. Each entry in the left part of the table represents a conditional probability, for example, the top-left entry represents $P(X_{\text{Project}} = 1|Y = 1)$. **Do not use any smoothing to regularize the probabilities.**

	$X_{\text{Project}} = 1$	$X_{\text{Vacation}} = 1$	$X_{\text{Deadline}} = 1$	$X_{\text{Relaxing}} = 1$	$P(Y)$
$Y = 1$	1	0.5	1	0	0.5
$Y = 0$	0.5	1	0.5	0.5	0.5

After you estimate the parameters of the Naive Bayes classifier, use it to classify the following test sample:

X_{Project}	X_{Vacation}	X_{Deadline}	X_{Relaxing}
1	1	1	1

Solution:

$$\begin{gathered} P(Y=1)=0.5, \quad P(Y=0)=0.5, \\ P(X_{\text{Project}}=1|Y=1)=1, \quad P(X_{\text{Vacation}}=1|Y=1)=0.5, \\ P(X_{\text{Deadline}}=1|Y=1)=1, \quad P(X_{\text{Relaxing}}=1|Y=1)=0, \\ P(X_{\text{Project}}=1|Y=0)=0.5, \quad P(X_{\text{Vacation}}=1|Y=0)=1, \\ P(X_{\text{Deadline}}=1|Y=0)=0.5, \quad P(X_{\text{Relaxing}}=1|Y=0)=0.5. \end{gathered}$$

For $Y=1$,

$$\begin{aligned} & P(Y=1, X_{\text{Project}}=1, X_{\text{Vacation}}=1, X_{\text{Deadline}}=1, X_{\text{Relaxing}}=1) \\ = & P(Y=1)\cdot P(X_{\text{Project}}=1|Y=1)\cdot P(X_{\text{ProVacationject}}=1|Y=1) \\ & \cdot P(X_{\text{Deadline}}=1|Y=1)\cdot P(X_{\text{Relaxing}}=1|Y=1)=0. \end{aligned}$$

For $Y=0$,

$$\begin{aligned} & P(Y=0, X_{\text{Project}}=1, X_{\text{Vacation}}=1, X_{\text{Deadline}}=1, X_{\text{Relaxing}}=1) \\ = & P(Y=0)\cdot P(X_{\text{Project}}=1|Y=0)\cdot P(X_{\text{ProVacationject}}=1|Y=0) \\ & \cdot P(X_{\text{Deadline}}=1|Y=0)\cdot P(X_{\text{Relaxing}}=1|Y=0)=\frac{1}{16}. \end{aligned}$$

Thus

$$Y=\operatorname*{argmax}_{y} P(Y=y, X_{\text{Project}}=1, X_{\text{Vacation}}=1, X_{\text{Deadline}}=1, X_{\text{Relaxing}}=1)=0.$$

4.2 Unsupervised Naive Bayes (5 pts)

Mike lost his annotation of the training data and do not want to annotate them again. As shown in the following table, Mike has unlabeled training data. Now he wants to use the EM algorithm to learn the classifier unsupervisedly.

Y	X_{Project}	X_{Vacation}	X_{Deadline}	X_{Relaxing}
?	1	0	1	0
?	1	1	0	0
?	1	1	1	0
?	0	1	1	1

Recall the EM algorithm:

- We initialize the model parameters randomly.
- We keep applying E-step and M-step until convergence.

Mike initializes the model parameters as shown in the following table.

	$X_{\text{Project}} = 1$	$X_{\text{Vacation}} = 1$	$X_{\text{Deadline}} = 1$	$X_{\text{Relaxing}} = 1$	$P(Y)$
$Y = 1$	1	0.5	0.5	0	0.5
$Y = 0$	0	0.5	0.5	1	0.5

Help Mike apply the first E-step. For each data sample D_i (a row in the following table), calculate its label distribution given the parameters θ and D_i. Fill the blanks in the following table.

$P(Y = 1\|\theta, D_i)$	X_{Project}	X_{Vacation}	X_{Deadline}	X_{Relaxing}
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0
0	0	1	1	1

Use the results you just computed to run the first M-step. Update all the model parameters and fill the following table:

	$X_{\text{Project}} = 1$	$X_{\text{Vacation}} = 1$	$X_{\text{Deadline}} = 1$	$X_{\text{Relaxing}} = 1$	$P(Y)$
$Y = 1$	1	2/3	2/3	0	0.75
$Y = 0$	0	1	1	1	0.25

Solution:

$$
\begin{aligned}
&P(Y = 1|\theta, D_1) \propto 0.5 \times (1 \times 0.5 \times 0.5 \times 1),\\
&P(Y = 0|\theta, D_1) \propto 0.5 \times (0 \times 0.5 \times 0.5 \times 0),\\
&P(Y = 1|\theta, D_1) = 1.\\
&P(Y = 1|\theta, D_2) \propto 0.5 \times (1 \times 0.5 \times 0.5 \times 1),\\
&P(Y = 0|\theta, D_2) \propto 0,\\
&P(Y = 1|\theta, D_2) = 1.\\
&P(Y = 1|\theta, D_3) \propto 0.5 \times (1 \times 0.5 \times 0.5 \times 1),\\
&P(Y = 0|\theta, D_3) \propto 0,\\
&P(Y = 1|\theta, D_3) = 1.\\
&P(Y = 1|\theta, D_4) \propto 0.5 \times (0 \times 0.5 \times 0.5 \times 0) = 0,\\
&P(Y = 0|\theta, D_4) \propto 0.5 \times (1 \times 0.5 \times 0.5 \times 1),\\
&P(Y = 1|\theta, D_4) = 0.
\end{aligned}
$$

5 Probabilistic Context-Free Grammar (10 pts)

You are given a sentence: *book the flight to Shanghai*. Consider the following probabilistic CFG. Words in italics are terminals.

S → NP VP	0.5	NP → *Shanghai*	0.2
S → AUX NP VP	0.2	NP → Det Nominal	0.1
S → V NP	0.2	Nominal → *book*	0.4
S → VP PP	0.1	Nominal → *flight*	0.2
VP → A	0.5	Nominal → Nominal PP	0.4
VP → B	0.5	Det → *the*	0.5
B → C	1.0	Det → *a*	0.2
A → V NP	1.0	Det → *an*	0.3
C → VP PP	1.0	PP → P NP	1.0
V → *book*	0.5	P → *through*	0.5
V → *include*	0.5	P → *to*	0.5
NP → *Beijing*	0.7		

5.1 Convert to Chomsky Normal Form (3 pts)

Convert the grammar above to the Chomsky normal form. Write down all the production rules with their probabilities in the new grammar.

Solution:

Converted rules with probabilities			
S → NP VP	0.5	Nominal → *book*	0.4
S → AUX X	0.2	Nominal → *flight*	0.2
X → NP VP	1.0	NP → Det Nominal	0.1
S → V NP	0.2	Nominal → Nominal PP	0.4
S → VP PP	0.1	Det → *the*	0.5
VP → V NP	0.5	Det → *a*	0.2
VP → VP PP	0.5	Det → *an*	0.3
V → *book*	0.5	PP → P NP	1.0
V → *include*	0.5	P → *through*	0.5
NP → *Shanghai*	0.2	P → *to*	0.5
NP → *Beijing*	0.7		

5.2 CYK (4 pts)

Use the CYK algorithm to find the parse tree with the highest probability. Remember to write down both the nonterminals and the corresponding probabilities in the table. Use → to connect the parents and children in the best parse tree.

book	the	flight	to	Shanghai
V: 0.5 Nominal: 0.4		VP: 0.0025 S: 0.001		S: 0.00004 VP: 0.000125
	Det: 0.5	NP: 0.01		NP: 0.0004
		Nominal: 0.2		Nominal: 0.008
			P: 0.5	PP: 0.1
				NP: 0.2

5.3 All the Parse Trees (3 pts)

Draw all the parse trees of the sentence given the grammar in the Chomsky normal form.

Solution: