CS 4780/5780: Machine Learning for Intelligent Systems **(Due: 11/26/19)**

Assignment #5: Naive Bayes, LDA, Viterbi, Online Learning

Instructor: Nika Haghtalab, Thorsten Joachims *Name:* Student name(s)*, Netid:* NetId(s)

Course Policy: Read all the instructions below carefully before you start working on the assignment, and before you make a submission.

- Please include your name and NetIDs on the first page. We recommend typesetting your submission in LaTeX, and an Overleaf template is linked here.
- Assignments are due by noon on the due date in PDF form on Gradescope.
- Late assignments can be submitted on Gradescope up to noon Sunday, Dec 1. This is also when the solutions will be released.
- You can do this assignment in groups of 2-3. Please submit no more than one submission per group. Collaboration across groups is not permitted.
- All sources of material outside the course must be cited. The University Academic Code of Conduct will be strictly enforced.

Submission Instructions: All group members must be added to the Gradescope submission. If you're the one submitting, add your group members on Gradescope. Otherwise, make sure you are added to the submission. Put your names on the PDF, this helps us track the groups in case there are errors on Gradescope. See this Piazza post for more information.

Problem 1: Naive Bayes

(10 + 10 = 20 points)

Consider the problem of classifying examples with two labels $Y = \{-1, 1\}$ and three binary features X_1, X_2, X_3. The examples are drawn from a distribution with

- $P(Y = -1) = P(Y = 1) = 0.5$
- $P(X_1 = 0|Y = -1) = 0.8$, $P(X_1 = 0|Y = 1) = 0.2$
- $P(X_i = 0|Y = -1) = 0.3$, $P(X_i = 0|Y = 1) = 0.7$ for $i = 2, 3$.
- X_1 independent of X_2, X_3 given the class label Y.

(a) If in addition X_2 and X_3 are assumed independent given Y, how would a naive Bayes classifier label point $x = (1, 1, 1)$? Show your work, especially the values of $P(X = x, Y = y)$ for both classes.

(b) What is the Bayes-optimal labeling of $x = (1, 1, 1)$ if X_2 and X_3 are not independent, but fully dependent with $X_2 = X_3$. Show your work, especially the values of $P(X = x, Y = y)$ for both classes.

Problem 2: LDA Decision Boundaries

(10 + 10 = 20 points)

LDA with diagonal covariance and equal variance of 1 classifies points in $\mathbb{R}^p$ according to the decision function:

$$h(x) = \arg\max_{y_i \in Y} P(Y = y_i) \cdot (2\pi)^{-p/2} e^{-\frac{1}{2}(x-\mu_i)^\top (x-\mu_i)}$$

The parameters $P(Y = y)$ and μ_i are typically estimated from a training set. Notice that while the classes have distinct means, the features are assumed to vary identically in each class.

(a) Express the decision function $h(x)$ for 2-class LDA as a linear classifier of the form $h(x) = sign(w \cdot x + b)$. What are w, b?

(b) Now, suppose the variance differs between classes. Assume the decision function takes the form of $h(x) = sign(f(x))$. What is $f(x)$? What shape is the boundary? In answering this question, you may assume x is a scalar so that $h(x)$ simplifies to:

$$h(x) = \arg\max_{y_i \in Y} P(Y = y_i) \cdot (2\pi\sigma_i)^{-1/2} \cdot e^{-\frac{1}{2\sigma_i}(x-\mu_i)^2}$$

Problem 3: Viterbi Algorithm

(15 + 7 + 8 = 30 points)

We learned Hidden Markov Models (HMMs) as a generative model, and the Viterbi Algorithm is used to compute the most likely configuration of the hidden variables. HMMs are useful for various tasks such as Speech Recognition, Machine Translation, and Signal Encoding, and consequently the Viterbi algorithm is highly important technique in Machine Learning. In this assignment, we will investigate a code translation problem that can also be solved by the Viterbi algorithm.

In the beginning of the semester, Ian created a simple linguistic code that utilizes several Greek alphabets in order to safely encrypt his password. After a few months, he forgets the original English password, and only remembers the Greek encryption. He believes that strong similarity between English and Greek alphabets was incorporated for his encryption, whereas there was no deterministic mapping between English and Greek characters. Thus we hypothesize that sequences of Greek characters correspond to English words and plan to build an HMM to help him decipher his password.

Formally speaking, we represent a Greek word as $x = (x_1, x_2, \dots x_T)$ where $x_t \in \{\alpha, \tau, \eta, \gamma, \omega\}$ is the t-th character in the word (represented in Greek script). Given each Greek word, our goal is to predict the most probable English word $y = (y_1, y_2, \dots, y_T)$ where $y_t \in \{a, i, p, s\}$ is the English translation of the t-th character (represented in English script). The following tables give the transition and emission probabilities of our HMM.

	a	i	p	s
a	0.05	0.1	0.15	0.7
i	0.1	0.05	0.25	0.6
p	0.45	0.15	0.05	0.35
s	0.35	0.2	0.15	0.3
Start	0.1	0.4	0.2	0.3

Table 1: Transition Probabilities $P(y_t|y_{t-1})$ where y_{t-1} on each row, y_t on each column

	α	τ	η	γ	ω
a	0.4	0.2	0.1	0.2	0.1
i	0.3	0.1	0.4	0.1	0.1
p	0.1	0.1	0.1	0.2	0.5
s	0.1	0.4	0.1	0.3	0.1

Table 2: Emission Probabilities $P(x_t|y_t)$ where y_t on each row, x_t on each column

In HMMs, the transition probability $P(y_t, |y_{t-1})$ decides the probability of the current English character given the previous English character, whereas the emission probability $P(x_t|y_t)$ determines the probability of a Greek translation given a English character. Based on these two model probabilities, we can compute the most likely English translation of each Greek word $x = (x_1, x_2, \dots, x_T)$ via the following HMM formula:

$$\arg\max_{y_1, y_2, \dots, y_T} P(y_1, y_2, \dots, y_T | x_1, x_2, \dots, x_T) = \arg\max_{y_1, y_2, \dots y_T} P(y_1) P(x_1|y_1) \Pi_{t=2}^{T} P(x_t|y_t) P(y_t|y_{t-1})$$

Let $\delta_{s,t}$ be the probability of the most probable English sequence corresponding to the first t Greek observations $(x_1, x_2, \dots, x_t)$ that end with the English character s. By definition, $\delta_{y,t-1}$ is the probability of the most probable English sequence corresponding to the first $t-1$ Greek observations that end with the English character y. So, the most probable sequence including the next observation x_t corresponding to English character s is given by finding the best transition from any y to s and the emission from s to x_t. For $s \in \{a, i, p, s\}$ and $2 \le t \le T$ the recurrence relation is

$$\delta_{s,t} = P(X_t = x_t | Y_t = s) \cdot \max_{y \in \{a,i,p,s\}} P(Y_t = s | Y_{t-1} = y) \delta_{y,t-1}$$

The initial condition is

$$\delta_{s,1} = P(X_1 = x_1|Y_1 = s) \cdot P_{Start}(s)(s \in \{a, i, p, s\})$$

(a) Ian's encrypted password is "$\omega\alpha\gamma\tau$ $\eta\tau$ $\gamma\alpha\omega$". What are the most likely English translations for each of the Greek encrypted words: 1) $\eta\tau$, 2) $\gamma\alpha\omega$, 3) $\omega\alpha\gamma\tau$? For each Greek word, fill out the 2D dynamic programming tables (conventionally s will vary on the row side and t will vary on the column side of the table) whose entries are $\delta_{s,t}$ with specifying the backtracking path that corresponds to the most probably translation. Table templates are included in the latex template as well. Replace '0.0' and 'x' in the tables with the numbers and letters from your calculations. What is Ian's English password?

$\delta_{s,t}$	$t = 1$	$t = 2$
a	0.0	0.0 (from x)
i	0.0	0.0 (from x)
p	0.0	0.0 (from x)
s	0.0	0.0 (from x)

Table 3: $\delta_{s,t}$ for $\eta\tau$

$\delta_{s,t}$	$t = 1$	$t = 2$	$t = 3$
a	0.0	0.0 (from x)	0.0 (from x)
i	0.0	0.0 (from x)	0.0 (from x)
p	0.0	0.0 (from x)	0.0 (from x)
s	0.0	0.0 (from x)	0.0 (from x)

Table 4: $\delta_{s,t}$ for $\gamma\alpha\omega$

$\delta_{s,t}$	$t = 1$	$t = 2$	$t = 3$	$t = 4$
a	0.0	0.0 (from x)	0.0 (from x)	0.0 (from x)
i	0.0	0.0 (from x)	0.0 (from x)	0.0 (from x)
p	0.0	0.0 (from x)	0.0 (from x)	0.0 (from x)
s	0.0	0.0 (from x)	0.0 (from x)	0.0 (from x)

Table 5: $\delta_{s,t}$ for $\omega\alpha\gamma\tau$

(b) Suppose English has m characters and Greek has n characters in their alphabets. What is the running time of the translation from a Greek word of length T into an English word via the Viterbi algorithm? Compare it to the running time of the brute-force algorithm naively considering all possible English letter sequences in terms of big-O complexity. (You could assume reading the model probabilities takes only a constant time)

(c) Given the transition and emission probabilities of a first-order HMM model, describe how to compute the probability of a Greek word $x = (x_1, x_2, \ldots, x_T)$ (ie $P(X = (x_1, x_2, \ldots, x_T))$). You have to clearly specify the equations you formulate for calculating this probability as well as a precise 2-4 sentence description of how your algorithm would work. (note: Solutions with exponential time complexity will get the full credits, but a sub-exponential time algorithm does exist as well)

Problem 4: Online Learning (10 + 10 + 10 = 30 points)

(a) Consider instance space $X = \{1, 2, \ldots, n\}$, for any $n > 3$, and hypothesis class $H = \{h_j | j \in [n]\}$ such that $h_j(i) = 1$ if $i = j$ and $h_j(i) = -1$ otherwise. Suppose the Halving learning algorithm is presented with the infinite sequence of inputs $\{1, 2, \ldots, n, 1, \ldots, n, \ldots\}$ with revealed labels $l(i) = 1$ for $i = 2$ and $l(i) = -1$ for $i \neq 2$. How many mistakes are made by the Halving algorithm on this infinite sequence?

(b) Suppose the Weighted Majority algorithm is run on the above problem where $n = 5$, and with update parameter $\beta = \frac{1}{2}$. Fill in the table for the algorithm on the sequence $\{1, 2, 3, 4, 5, 1, 2, 3\}$ with labels $l(i) = 1$ for $i = 2$ and $l(i) = -1$ for $i \neq 2$. Row t in this table should show the following information: the set of weights w_j corresponding to the weight of hypothesis h_j that is used for predicting the label of the t^{th} instance, the t^{th} instance and its label, the algorithm's prediction for the t^{th} instance, and whether the algorithm makes a mistake on this instance. The first line is filled out for you.

w_1	w_2	w_3	w_4	w_5	Input	Label	Prediction	Mistake (Y/N)
1	1	1	1	1	1	-1	-1	N
					2	1		
					3	-1		
					4	-1		
					5	-1		
					1	-1		
					2	1		
					3	-1		

Table 6: 4b

(c) Now suppose the Weighted Majority algorithm is run on the same problem where $n = 2^{2k} + 1$ for some integer $k \geq 1$, and with update parameter $\beta = \frac{1}{2}$. How many mistakes are made on the infinite sequence $\{1, 2, \ldots, n, 1, 2, \ldots, n, \ldots\}$ with revealed labels $l(i) = 1$ for $i = 2$ and $l(i) = -1$ for $i \neq 2$?