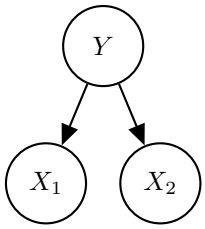
CS 188 Fall 2020 Introduction to Artificial Intelligence **Final Review ML Solution**

Q1. Naïve Bayes

You are given a naïve bayes model, shown below, with label Y and features X_1 and X_2. The conditional probabilities for the model are parametrized by p_1, p_2 and q.



X_1	Y	$P(X_1\|Y)$
0	0	p_1
1	0	$1-p_1$
0	1	$1-p_1$
1	1	p_1

X_2	Y	$P(X_2\|Y)$
0	0	p_2
1	0	$1-p_2$
0	1	$1-p_2$
1	1	p_2

Y	$P(Y)$
0	$1-q$
1	q

Note that some of the parameters are shared (e.g. $P(X_1 = 0|Y = 0) = P(X_1 = 1|Y = 1) = p_1$).

(a) Given a new data point with $X_1 = 1$ and $X_2 = 1$, what is the probability that this point has label $Y = 1$? Express your answer in terms of the parameters p_1, p_2 and q (you might not need all of them).

$P(Y = 1|X_1 = 1, X_2 = 1) =$ $\frac{p_1p_2q}{p_1p_2q+(1-p_1)(1-p_2)(1-q)}$

$$
\begin{aligned}
P(Y = 1, X_1 = 1, X_2 = 1) &= P(X_1 = 1|Y = 1)P(X_2 = 1|Y = 1)P(Y = 1) \\
&= p_1p_2q \\
P(Y = 0, X_1 = 1, X_2 = 1) &= P(X_1 = 1|Y = 0)P(X_2 = 1|Y = 0)P(Y = 0) \\
&= (1-p_1)(1-p_2)(1-q) \\
P(Y = 1|X_1 = 1, X_2 = 1) &= \frac{P(Y = 1, X_1 = 1, X_2 = 1)}{P(X_1 = 1, X_2 = 1)} \\
&= \frac{P(Y = 1, X_1 = 1, X_2 = 1)}{P(Y = 1, X_1 = 1, X_2 = 1) + P(Y = 0, X_1 = 1, X_2 = 1)} \\
&= \frac{p_1p_2q}{p_1p_2q + (1-p_1)(1-p_2)(1-q)}
\end{aligned}
$$

The model is trained with the following data:

sample number	1	2	3	4	5	6	7	8	9	10
X_1	0	0	1	0	1	0	1	0	1	1
X_2	0	0	0	0	0	0	0	1	0	0
Y	0	0	0	0	0	0	0	1	1	1

(b) What are the maximum likelihood estimates for p_1, p_2 and q?

$p_1 =$ $\frac{3}{5}$ $p_2 =$ $\frac{4}{5}$ $q =$ $\frac{3}{10}$

The maximum likelihood estimate of p_1 is the fraction of counts of samples in which $X_1 = Y$. In the given training data, samples $1, 2, 4$ and 6 have $X_1 = Y = 0$ and samples 9 and 10 have $X_1 = Y = 1$, so 6 out of the 10 samples have $X_1 = Y$ and thus $p_1 = \frac{6}{10} = \frac{3}{5}$. Analogously, 8 out of the 10 samples have $X_2 = Y$ and thus $p_2 = \frac{8}{10} = \frac{4}{5}$. The maximum likelihood estimate of q is the fraction of counts of samples in which $Y = 1$, thus $q = \frac{3}{10}$.

Q2. Machine Learning: Potpourri

(a) What it the **minimum** number of parameters needed to fully model a joint distribution $P(Y, F_1, F_2, ..., F_n)$ over label Y and n features F_i? Assume binary class where each feature can possibly take on k distinct values.

$2k^n - 1$

(b) Under the **Naive Bayes assumption**, what is the **minimum** number of parameters needed to model a joint distribution $P(Y, F_1, F_2, ..., F_n)$ over label Y and n features F_i? Assume binary class where each feature can take on k distinct values.

$2n(k-1)+1$

(c) You suspect that you are overfitting with your Naive Bayes with Laplace Smoothing. How would you adjust the strength k in Laplace Smoothing?

- ● Increase k
- ○ Decrease k

(d) While using Naive Bayes with Laplace Smoothing, increasing the strength k in Laplace Smoothing can:

- ■ Increase training error
- ☐ Decrease training error
- ■ Increase validation error
- ■ Decrease validation error

(e) It is possible for the perceptron algorithm to never terminate on a dataset that is linearly separable in its feature space.

- ○ True
- ● False

(f) If the perceptron algorithm terminates, then it is guaranteed to find a max-margin separating decision boundary.

- ○ True
- ● False

(g) In multiclass perceptron, every weight w_y can be written as a linear combination of the training data feature vectors.

- ● True
- ○ False

(h) For binary class classification, logistic regression produces a linear decision boundary.

- ● True
- ○ False

(i) In the binary classification case, logistic regression is exactly equivalent to a single-layer neural network with a sigmoid activation and the cross-entropy loss function.

● True ○ False

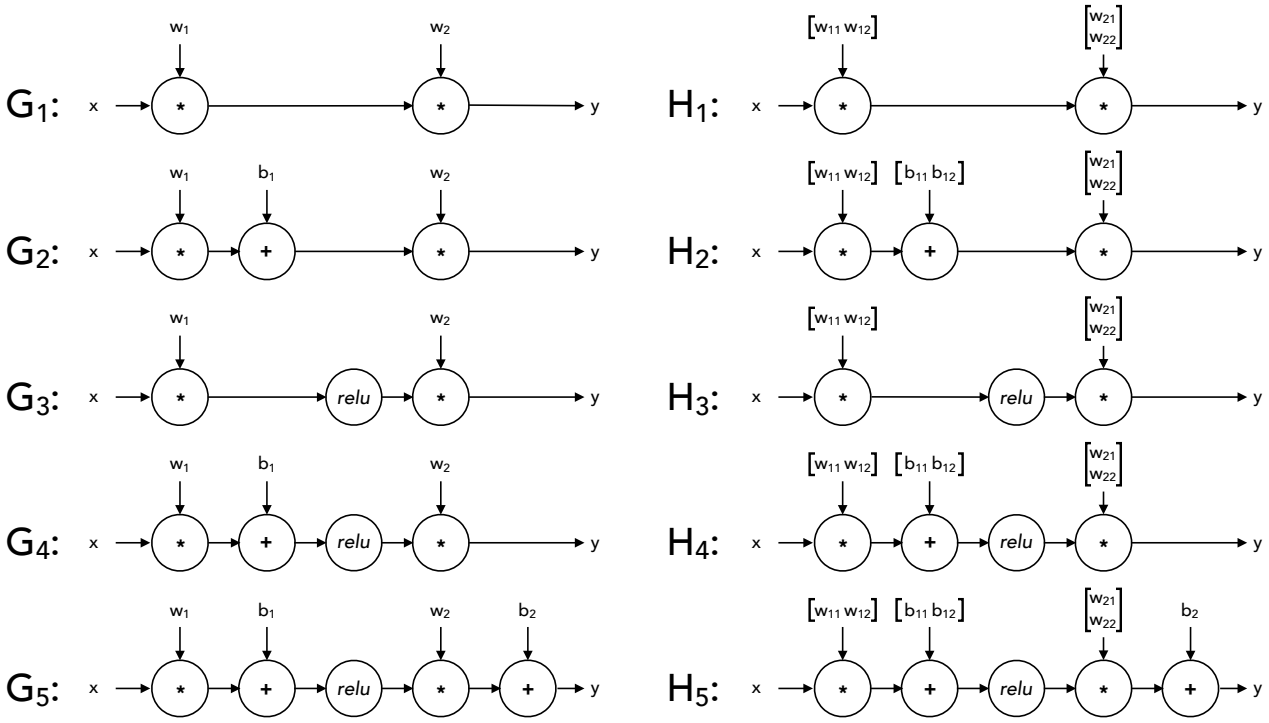
(j) **(i)** You train a linear classifier on 1,000 training points and discover that the training accuracy is only 50%. Which of the following, if done in isolation, has a good chance of improving your training accuracy?

■ Add novel features ☐ Train on more data ■ Train on less data

(ii) You now try training a neural network but you find that the training accuracy is still very low. Which of the following, if done in isolation, has a good chance of improving your training accuracy?

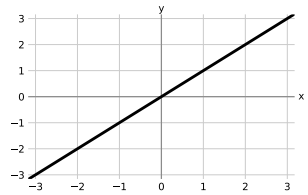
■ Add more hidden layers ■ Add more units to the hidden layers

Q3. Neural Networks: Representation

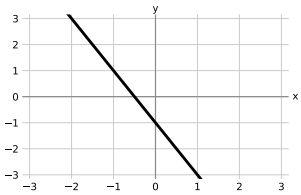


For each of the piecewise-linear functions below, mark all networks from the list above that can represent the function **exactly** on the range $x \in (-\infty, \infty)$. In the networks above, *relu* denotes the element-wise ReLU nonlinearity: $relu(z) = max(0, z)$. The networks G_i use 1-dimensional layers, while the networks H_i have some 2-dimensional intermediate layers.

(a)



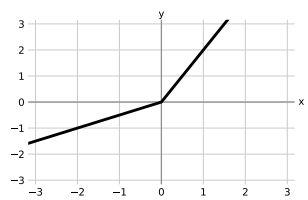
- [x] G_1 [x] H_1
- [x] G_2 [x] H_2
- [ ] G_3 [x] H_3
- [ ] G_4 [x] H_4
- [ ] G_5 [x] H_5
- ○ None of the above



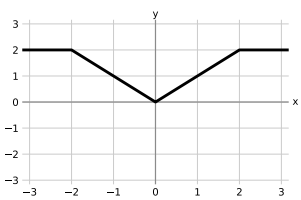
- [ ] G_1 [ ] H_1
- [x] G_2 [x] H_2
- [ ] G_3 [ ] H_3
- [ ] G_4 [x] H_4
- [ ] G_5 [x] H_5
- ○ None of the above

The networks G_3, G_4, G_5 include a ReLU nonlinearity on a scalar quantity, so it is impossible for their output to represent a non-horizontal straight line. On the other hand, H_3, H_4, H_5 have a 2-dimensional hidden layer, which allows two ReLU elements facing in opposite directions to be added together to form a straight line. The second subpart requires a bias term because the line does not pass through the origin.

(b)



- [ ] G_1 [ ] H_1
- [ ] G_2 [ ] H_2
- [ ] G_3 [x] H_3
- [ ] G_4 [x] H_4
- [ ] G_5 [x] H_5
- ○ None of the above



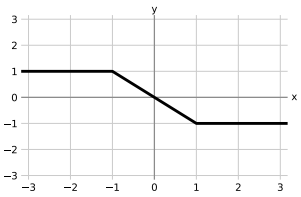
- [ ] G_1 [ ] H_1
- [ ] G_2 [ ] H_2
- [ ] G_3 [ ] H_3
- [ ] G_4 [ ] H_4
- [ ] G_5 [ ] H_5
- ● None of the above

These functions include multiple non-horizontal linear regions, so they cannot be represented by any of the networks G_i which apply ReLU no more than once to a scalar quantity.

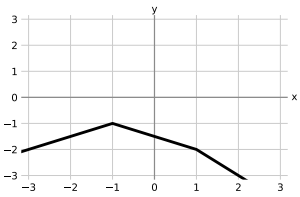
The first subpart can be represented by any of the networks with 2-dimensional ReLU nodes. The point of nonlinearity occurs at the origin, so nonzero bias terms are not required.

The second subpart has 3 points where the slope changes, but the networks H_i only have a single 2-dimensional ReLU node. Each application of ReLU to one element can only introduce a change of slope for a single value of x.

(c)



☐ G_1 ☐ H_1
☐ G_2 ☐ H_2
☐ G_3 ☐ H_3
☐ G_4 ☐ H_4
☐ G_5 ■ H_5
◯ None of the above



☐ G_1 ☐ H_1
☐ G_2 ☐ H_2
☐ G_3 ☐ H_3
☐ G_4 ■ H_4
☐ G_5 ■ H_5
◯ None of the above

Both functions have two points where the slope changes, so none of the networks $G_i; H_1, H_2$ can represent them.

An output bias term is required for the first subpart because one of the flat regions must be generated by the flat part of a ReLU function, but neither one of them is at $y = 0$.

The second subpart doesn't require a bias term at the output: it can be represented as $-relu(\frac{-x+1}{2}) - relu(x + 1)$. Note how if the segment at $x > 2$ were to be extended to cross the x axis, it would cross exactly at $x = -1$, the location of the other slope change. A similar statement is true for the segment at $x < -1$.