COMS W4701: Artificial Intelligence, Spring 2024

Homework 6

Instructions: Compile all written solutions for this assignment in a single, typed PDF file. If you use Python or any other computational tool, append the code to your PDF file. Turn in your submission on Gradescope, and **tag all pages**. For Problem 3, append a printout to your PDF file AND separately upload the completed Jupyter notebook to Gradescope. Please be mindful of the deadline and late policy, as well as our policies on citations and academic honesty.

We will be working with the following data set for Problems 1 and 2. There are two trinary features x_1 and x_2, and a class variable y with values 0 and 1.

Sample	x_1	x_2	y
1	$+1$	-1	0
2	$+1$	$+1$	0
3	0	-1	0
4	0	$+1$	0
5	$+1$	$+1$	1
6	0	0	1
7	0	$+1$	1
8	-1	$+1$	1

Problem 1: Naïve Bayes (20 points)

1. The maximum likelihood class CPT is $\Pr(Y) = (0.5, 0.5)$. Estimate the feature CPTs $\Pr(X_1|Y)$ and $\Pr(X_2|Y)$ without Laplace smoothing.
2. Which value(s) of x_1 give a direct prediction of y regardless of the value of x_2, and what are the predictions of y for each? Give the same analysis for values of x_2. Are there any feature combinations (x_1, x_2) for which a prediction is not well defined?
3. Suppose that we find out that the class values y in the data set may not be correct. We can use the estimated probabilities in Part 1 as a starting point for expectation-maximization. Compute the expected counts $\Pr(Y|x_1, x_2)$ for each sample (x_1, x_2) in the data set.
4. Perform the maximization step and find the new CPTs $\Pr(Y)$, $\Pr(X_1|Y)$, and $\Pr(X_2|Y)$.

Problem 2: Linear Classifiers (20 points)

1. Suppose we drop samples 4 and 5, leaving us with six data points. We want to learn a linear classifier $f_{\mathbf{w}}$ such that we predict 0 when $f_{\mathbf{w}} \leq 0$ and 1 otherwise. Starting with the weight vector $\mathbf{w} = (1, -2, 0)$, use the perceptron learning rule and learning rate $\alpha = 1$ to process the six data points once in order. Show the result of each sample classification and any weight updates that occur.

2. Plot the data in x_1-x_2 space (you can either hand-draw or use Python). Use different indicators for each class. Then overlay the decision boundaries from the original weight vector $\mathbf{w} = (1, -2, 0)$ and the new weight vector $\mathbf{w}'$ that you found in Part 1. Did perceptron find a model that performs perfect classification? Would you say that the model is a good one in terms of separability?

3. Now use sklearn to fit a logistic model to the original data set. Show a DecisionBoundaryDisplay() plot overlaid on a scatter plot of the data. What are the learned weights and classification accuracy on the data set?

4. Compute the mean negative log likelihood of the data given the learned weights. (You can either apply the formula directly or use a function like `predict_log_proba()`.) Compute the gradient of the negative log likelihood, evaluated at the current weight vector.

Problem 3: Image Classification (60 points)

In this problem, you will implement a convolutional neural network to classify images in imagenette, a data set with about 13000 images across 10 different classes. We provide a notebook containing some import and setup code. You should download and then upload it to Google Colab on Google Drive. Then complete all specified tasks. For submission, append a printout to your PDF file AND separately upload the completed Jupyter notebook to Gradescope.