Homework 2: SOLUTIONS

Problem 1 (written) – 20 points

In this problem you will derive a naive Bayes classifier that you will implement in Problem 2 below. Recall that for a labeled set of data $(y_1, x_1), \ldots, (y_n, x_n)$, where for this problem $y \in \{0, 1\}$ and x is a D-dimensional vector not necessarily in $\mathbb{R}^D$, the Bayes classifier observes a new x_0 and predicts y_0 as

$$y_0 \;=\; \arg\max_y \; p(y_0 = y|\pi) \prod_{d=1}^{D} p_d(x_{0,d}|\theta_y^{(d)}).$$

The distribution $p(y_0 = y|\pi) = \text{Bernoulli}(y|\pi)$. What is "naive" about this classifier is the assumption that all D dimensions of x are independent. Observe that you can pick any distribution p_d you think appropriate for the dth dimension. In this problem, assume that $D = 2$ and p_1 is a Bernoulli distribution and p_2 is a Pareto distribution. That is,

$$p_1(x_{0,1}|\theta_y^{(1)}) = (\theta_y^{(1)})^{x_{0,1}}(1-\theta_y^{(1)})^{1-x_{0,1}}, \quad p_2(x_{0,2}|\theta_y^{(2)}) = \theta_y^{(2)}(x_{0,2})^{-(\theta_y^{(2)}+1)}.$$

The parameter $\theta_y^{(1)} \in [0, 1]$ and $\theta_y^{(2)} > 0$. For the class prior Bernoulli distribution, $\pi \in [0, 1]$. Given some data $(y_1, x_1), \ldots, (y_n, x_n)$, derive the maximum likelihood solution for π, $\theta_y^{(1)}$ and $\theta_y^{(2)}$, i.e.,

$$\widehat{\pi},\, \widehat{\theta}_y^{(1)},\, \widehat{\theta}_y^{(2)} \;=\; \arg \max_{\pi, \theta_y^{(1)}, \theta_y^{(2)}} \sum_{i=1}^{n} \ln p(y_i|\pi) + \sum_{i=1}^{n} \ln p(x_{i1}|\theta_{y_i}^{(1)}) + \sum_{i=1}^{n} \ln p(x_{i2}|\theta_{y_i}^{(2)}).$$

(I avoided copying θ for $y = 1$ and $y = 0$ above.) Notice that this can be viewed as three independent subproblems. Please separate your derivations as follows:

(a) Derive $\widehat{\pi}$ using the objective above.

SOLUTION: In the answer below, since $y_i \in \{0, 1\}$, all of the indicators may be replaced as follows: $\mathbf{1}(y_i = 1) \to y_i$ and $\mathbf{1}(y_i = 0) \to 1 - y_i$

$$L = \sum_{i=1}^{n} \ln p(y_i|\pi) = \sum_{i=1}^{n} \ln \mathbf{1}(y_i = 1) \ln \pi + \mathbf{1}(y_i = 0) \ln(1-\pi)$$

$$\frac{dL}{d\pi} = \sum_{i=1}^{n} \frac{\mathbf{1}(y_i = 1)}{\pi} - \frac{\mathbf{1}(y_i = 0)}{1-\pi} = 0$$

$$\pi = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(y_i = 1)$$

(b) Derive $\widehat{\theta}_y^{(1)}$ using the objective above. Derive this leaving y arbitrary.

SOLUTION: Below is the derivation for a generic value of y. Similar to Part (a), one can also use indicators: $x_{i1} \to \mathbf{1}(x_{i1} = 1)$ and $1 - x_{i1} = \mathbf{1}(x_{i1} = 0)$.

$$L = \sum_{i=1}^{n} \mathbf{1}(y_i = y) \ln p(x_{i1}|\theta_y^{(1)}) = \sum_{i=1}^{n} \mathbf{1}(y_i = y) x_{i1} \ln \theta_y^{(1)} + \mathbf{1}(y_i = y)(1 - x_{i1}) \ln(1 - \theta_y^{(1)})$$

$$\frac{dL}{d\theta_y^{(1)}} = \sum_{i=1}^{n} \frac{\mathbf{1}(y_i = y)x_{i1}}{\theta_y^{(1)}} + \frac{\mathbf{1}(y_i = y)(1 - x_{i1})}{1 - \theta_y^{(1)}} = 0$$

$$\theta_y^{(1)} = \frac{\sum_{i=1}^{n} \mathbf{1}(y_i = y)x_{i1}}{\sum_{i=1}^{n} \mathbf{1}(y_i = y)}$$

(c) Derive $\widehat{\theta}_y^{(2)}$ using the objective above. Derive this leaving y arbitrary.

SOLUTION:

$$L = \sum_{i=1}^{n} \mathbf{1}(y_i = y) \ln p(x_{i2}|\theta_y^{(2)}) = \sum_{i=1}^{n} \mathbf{1}(y_i = y) \ln \theta_y^{(2)} - \mathbf{1}(y_i = y)(\theta_y^{(2)} + 1) \ln x_{i2}$$

$$\frac{dL}{d\theta_y^{(2)}} = \sum_{i=1}^{n} \frac{\mathbf{1}(y_i = y)}{\theta_y^{(2)}} - \mathbf{1}(y_i = y) \ln x_{i2} = 0$$

$$\theta_y^{(2)} = \frac{\sum_{i=1}^{n} \mathbf{1}(y_i = y)}{\sum_{i=1}^{n} \mathbf{1}(y_i = y) \ln x_{i2}}$$

Problem 2 (coding) – 40 points

In this problem you will implement the naive Bayes classifier derived in Problem 1, as well as the k-NN algorithm and logistic regression algorithm. The data consists of examples of spam and non-spam emails, of which there are 4508 training examples and 93 testing examples. The feature vector x is a 57-dimensional vector extracted from the email and $y = 1$ indicates a spam email. The data has been preprocessed by me such that the first 54-dimensions of each observation is binary and the last three dimensions are positive numbers. As with the first homework, you would ideally use cross-validation on multiple partitions, but I am keeping it simple here with one training and one testing set.

For the naive Bayes classifier, use 54 Bernoulli distributions for the 54 binary dimensions and 3 Pareto distributions for the last 3 positive dimensions. I choose the Pareto distribution because it is able to model outliers more easily, which the data seems to have many of. There are other "heavy tail" distributions we could have chosen as well. (The reason for the Bernoulli distribution should be clear.)

(a) Implement the naive Bayes classifier described above on the training data and make predictions on the testing data. In a 2×2 table, write the number of times that you predicted a class y data point (ground truth) as a class y' data point (model prediction) in the (y, y')-th cell of the table, where y and y' can be either 0 or 1. There should be four values written in the table in your PDF. Next to your table, write the prediction accuracy—the sum of the diagonal divided by 93.

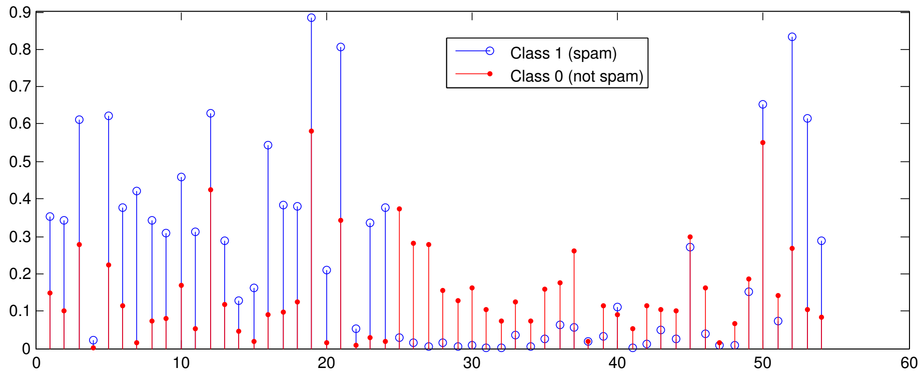
SOLUTION:

	$y' = 1$	$y' = 0$
$y = 1$	31	6
$y = 0$	2	54

Prediction accuracy is 0.914, or 91.4%

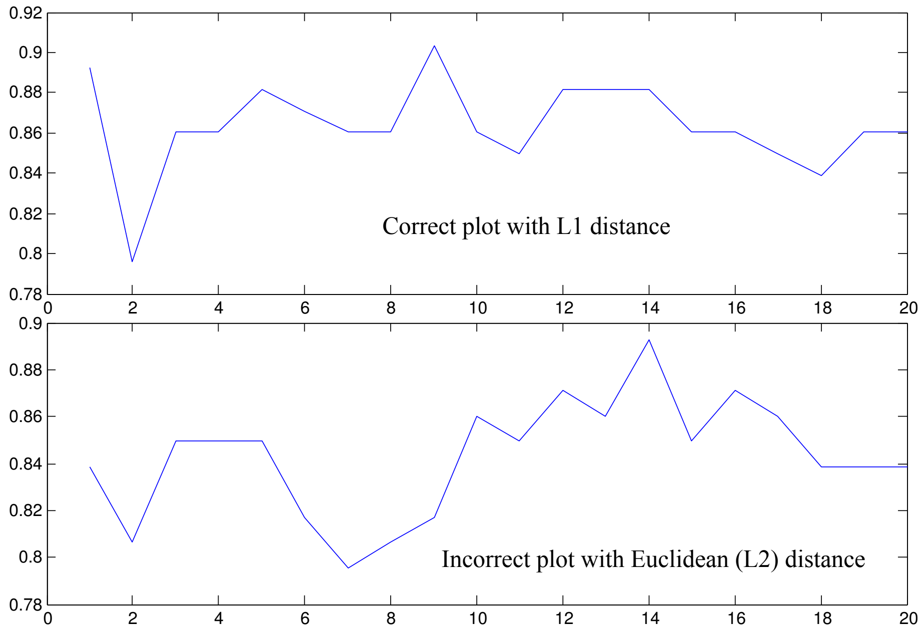
(b) In one figure, show a stem plot (`stem()` in Matlab) of the 54 Bernoulli parameters for each class. Use the file "spambase.names" to make an observation about dimensions 16 and 52.

SOLUTION: See plot below. Observations: Dimension 16 corresponds to the word "free." This word has a much higher probability of appearing in a spam email. Dimension 52 corresponds to the symbol "!" (exclamation point). This symbol also has a much higher probability of appearing in a spam email.



(c) Implement the k-NN algorithm for $k = 1, \ldots, 20$. Use the ℓ_1 distance for this problem (sum of the absolute values of the differences). Plot the prediction accuracy as a function of k.

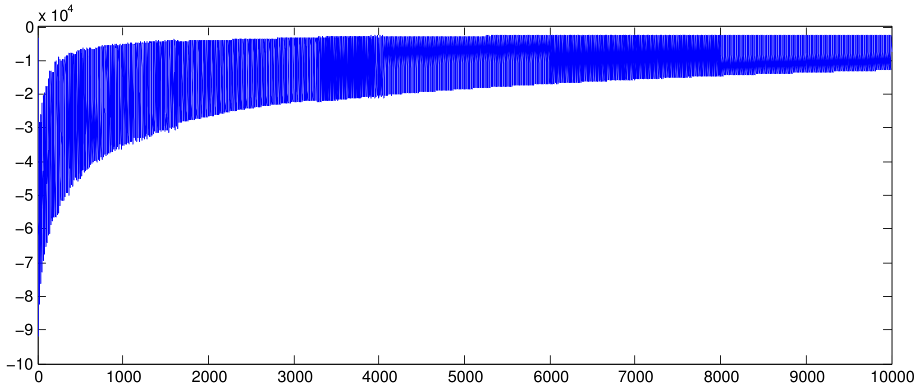
SOLUTION: See below. The plots may vary slightly because of ties in the data.



Finally, you will run logistic regression on the same data set. ***Set every*** $y_i = 0$ ***to*** $y_i = -1$ ***for this part. Also, be sure to add a dimension equal to*** $+1$ ***to each data point.***

(d) Implement the steepest ascent algorithm given in Lecture 9. Use an iteration-dependent step size η_t, where $\eta_t = \frac{1}{10^5\sqrt{t+1}}$. Run your algorithm for 10,000 iterations and plot the logistic regression objective training function $\mathcal{L}$ per iteration. (The pattern should look strange.)

SOLUTION:



(e) Finally, implement a gradient method called "Newton's method" as follows: At iteration t, set

$$w^{(t+1)} = w^{(t)} - \eta_t(\nabla_w^2\mathcal{L})^{-1}\nabla_w\mathcal{L}, \qquad \nabla_w^2\mathcal{L} = -\sum_{i=1}^{n}\sigma(x_i^Tw)(1-\sigma(x_i^Tw))x_ix_i^T \leftarrow \text{a matrix}$$

Set $\eta_t = \frac{1}{\sqrt{t+1}}$ for this problem. Plot the objective function $\mathcal{L}$ on the training data as a function of $t = 1, \ldots, 100$. Below the plot give the prediction accuracy on the testing data. We don't have time to go through Newton's method, but hopefully this will motivate you to study it more!

SOLUTION: The prediction accuracy is 0.914 or 91.4%.