COMS W4903: Machine Learning for Data Science

Columbia University, Spring 2017

Homework 2: Due February 26, 2017 by 11:59pm

Please read these instructions to ensure you receive full credit on your homework. Submit the written portion of your homework as a *single* PDF file through Courseworks (less than 5MB). In addition to your PDF write-up, submit all code written by you in their original extensions through Courseworks (e.g., .m, .r, .py, .c). Any coding language is acceptable, but do not submit notebooks, do not wrap your files in .rar, .zip, .tar and do not submit your write-up in .doc or other file type. Your grade will be based on the contents of one PDF file and the original source code. Additional files will be ignored. We will not run your code, so everything you are asked to show should be put in the PDF file. Show all work for full credit.

Late submission policy: Late homeworks will have 0.1% deducted from the final grade for each minute late. *Your homework submission time will be based on the time of your **last** submission to Courseworks. I will not revert to an earlier submission!* Therefore, do not re-submit after midnight on the due date unless you are confident the new submission is significantly better to overcompensate for the points lost. Submission time is non-negotiable and will be based on the time you submitted your last file to Courseworks. The number of points deducted will be rounded to the nearest integer.

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.

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

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

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.[1] 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.

(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.

(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.

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.)

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

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

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!

[1] The original data is at `https://archive.ics.uci.edu/ml/datasets/Spambase`. More information about the meanings of the 57 dimensions of the data is provided in two accompanying files.