COMS 4721 Spring 2016: Homework #1

Daniel Kost-Stephenson – dpk2124@columbia.edu

Discussants: Robert Minnich, Ryan Walsh

February 16, 2016

Problem 1

Problem 1 involved correctly classifying images of hand written digits from 0-9 using a 1-NN classifier. No library functions were used in the functions and the usage of for loops was minimized to improve the speed. The preds function uses numpy vector and matrix operations to compute Euclidean distances for each of the test points relative to the training points. The Euclidean distance calculations were performed as such:

$$\|\mathbf{p}-\mathbf{q}\| = \sqrt{(\mathbf{p}-\mathbf{q})\cdot(\mathbf{p}-\mathbf{q})} = \sqrt{\|\mathbf{p}\| + \|\mathbf{q}\| - 2\mathbf{p}\cdot\mathbf{q}}$$

$$\text{where} \quad \|\mathbf{p}\| = \sqrt{{p_1}^2 + {p_2}^2 + \cdots + {p_n}^2}$$

$$\text{and} \quad \|\mathbf{q}\| = \sqrt{{q_1}^2 + {q_2}^2 + \cdots + {q_n}^2}$$

$$\text{and} \quad 2\mathbf{p}\cdot\mathbf{q} = 2\sum_{i=1}^{n} p_i q_i$$

The output of the function is a $10000 \times n$ matrix, where $n \in \{1000, 2000, 4000, 8000\}$ is the number of training points utilized. The $10000 \times n$ matrix is actually a distance matrix, giving the Euclidean distance between any test point i, and any training point j and has the form: $\sqrt{\|\mathbf{p}_i\| + \|\mathbf{q}_j\| - 2\mathbf{p}_i\cdot\mathbf{q}_j}$. The index of the minimum in each row was taken, and the corresponding label of the training point was assigned to that test point. It is important to note that the size of the test data matrix and training data matrix do not match, so the dimensionality had to be adjusted so that the output was a $10000 \times n$ matrix. Ten random samples were taken for each of the four sizes of training points and a learning curve was plotted. The mean error rates of the random samples are displayed in the table below:

Table 1: Mean error rates and standard deviations for different sample sizes of training data.

Number of samples	Mean error rate	Standard deviation
1000	0.1147	0.0045

2000	0.0897	0.0022
4000	0.0692	0.002
8000	0.0562	0.0018

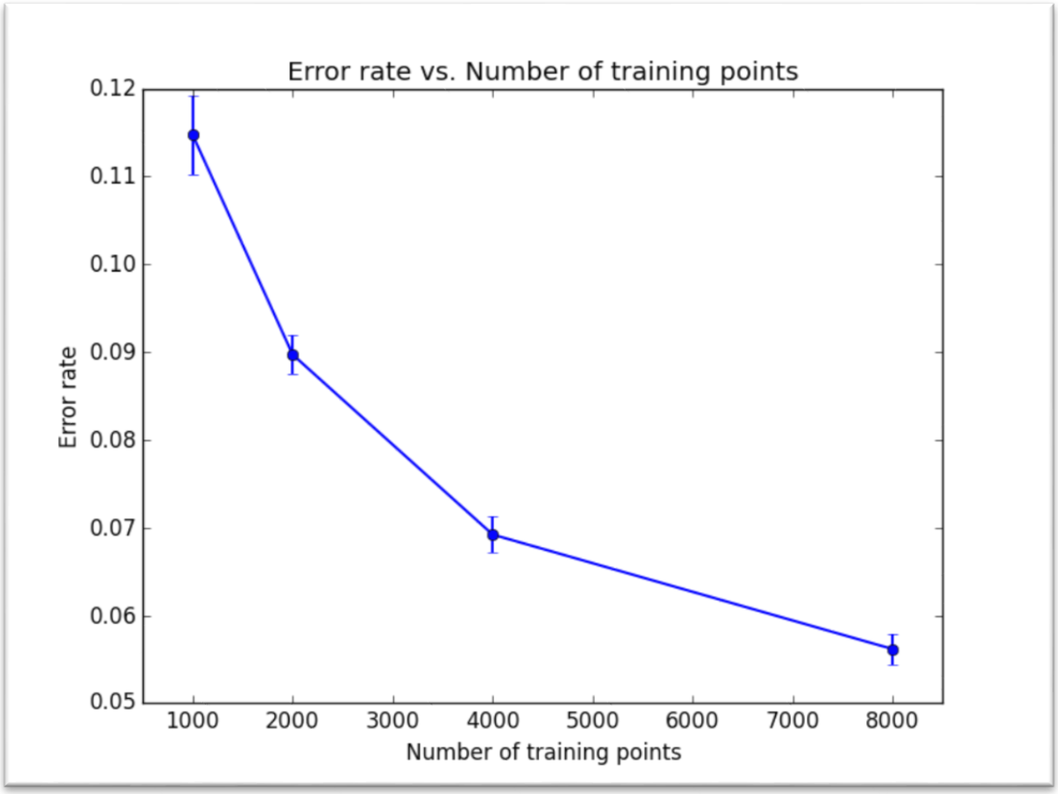

Figure 1: Learning curve of 1-NN classifier on ocr image data.

Problem 2

This problem involved classifying messages of diverse content to a set of 20 different news sources. A Naïve Bayes model was implemented to train and test the data. To estimate the parameters, a Laplace smoothing estimator was used to calculate the class conditional parameters.

Part a)

The MLE for this model can be formulated as such:

$$L(x;\mu_y) = \underset{y\,\epsilon\,\{1,2,\ldots,20\}}{\operatorname{argmax}} \rightarrow \pi_y \cdot P_{\mu_y}(x)$$

$$l(x;\mu_y) = \underset{y\,\epsilon\,\{1,2,\ldots,20\}}{\operatorname{argmax}} \rightarrow \log\left[\pi_y \cdot \prod_{j=1}^{d} \mu_{y,j}{}^{x_j}\left(1-\mu_{y,j}\right)^{(1-x_j)}\right]$$

Expanding the logs then taking the derivative and setting the above equation to zero, we obtain:

$$\sum_{j=1}^{d}\left(\frac{x_j}{\mu_{y,j}}-\frac{1-x_j}{1-\mu_{y,j}}\right)=0$$

The rearranging the terms, we obtain the estimate for the MLE:

$$\mu_{y,j}=\frac{1}{d}\sum_{j=1}^{d}x_j$$

Part b)

A Python function called `params` was implemented that took training data and training labels as input and returned a 20×61188 sized matrix of class conditional probabilities as output. A second function was implemented that took the parameters as input and returned predicted labels for the test set. The Laplace smoothing function was used to calculate class conditional probabilities, and the prior probabilities were calculated using $\pi_y = \frac{n_y}{N}$. For each test vector, the probability of that vector belonging to a given class was calculated using:

$$\widehat{p_y} = \underset{y\,\epsilon\,\{1,2,\dots,20\}}{\text{argmax}}\ \pi_y \cdot P_{\boldsymbol{\mu}}(\boldsymbol{x})$$

Where

$$P_{\boldsymbol{\mu}}(\boldsymbol{x}) = \prod_{j=1}^{d} \mu_{y,j}{}^{x_j}\left(1-\mu_{y,j}\right)^{(1-x_j)} \quad for\ \boldsymbol{x} = (x_1, x_2, \dots, x_d) \in \boldsymbol{\mathcal{X}}$$

The result was a 20×7505 matrix and the maximum value in each column was taken to determine the most likely label. A <u>training error rate of 0.0461</u> and a <u>test error rate of 0.2305</u> were obtained and are shown in the table below. It must be noted that multiplying the terms in the above equation together for a 61188 matrix causes the probability to equal 0 because the terms are so small. In response, a log transformation (logarithms are 1 to 1 functions and thus do not affect the results) was applied and a sum was taken instead of the product:

$$P_{\boldsymbol{\mu}}(\boldsymbol{x}) = \sum_{i=1}^{d} x_j \log\left(\mu_{y,j}\right) + \sum_{i=1}^{d}\left(1-x_j\right)\log\left(\mu_{y,j}\right)$$

Part c)

The most common words associated with each class are displayed in the table below. As is easily observed, most of the words are common words in the English language. It must be noted that some of the more unique words belonging to each class are still taken into account and are weighted accordingly. For example, if a certain class has unique vocabulary, the unique words are weighed more for that class then for the other classes. If a test vector contains some of these unique words, the likelihood of it belonging to the given class will be greater than the likelihood of it belonging to other classes. Because we are implementing a logarithm, a greater likelihood corresponds to less of a negative value and gives that specific class a greater chance of being the maximum.

Table 2: Most common words according to class

CLASS 1	if	on	but	edu	for	are	this	have	be	of
	not	and	that	you	the	in	is	writes	to	it
CLASS 2	any	if	with	or	can	but	you	be	have	of
	and	this	on	the	in	to	for	is	it	that
CLASS 3	can	be	if	edu	this	and	the	in	you	of
	to	for	have	it	that	on	but	with	is	windows
CLASS 4	but	you	be	if	my	or	can	this	with	of
	on	and	have	it	in	for	that	is	to	the
CLASS 5	can	not	you	edu	be	but	for	if	have	this
	that	with	on	it	of	to	is	in	and	the
CLASS 6	you	but	or	with	have	on	be	can	an	that
	if	this	is	it	for	of	to	in	the	and
CLASS 7	all	this	at	me	sale	it	if	are	edu	is
	or	you	have	with	of	for	to	in	the	and
CLASS 8	or	not	with	my	and	have	be	are	in	to
	you	for	of	car	it	is	writes	that	on	the
CLASS 9	but	have	com	this	dod	writes	that	article	and	of
	my	on	for	the	in	to	it	is	with	you
CLASS 10	he	at	on	for	edu	have	writes	with	it	this
	be	but	article	is	of	to	in	that	the	and
CLASS 11	are	they	was	with	have	and	this	for	the	of
	you	in	to	be	that	is	it	writes	but	on
CLASS 12	can	or	are	not	with	is	on	you	for	be
	this	that	writes	it	if	and	of	in	the	to
CLASS 13	can	if	but	with	are	or	you	that	have	for
	it	be	on	this	is	of	and	the	to	in
CLASS 14	on	edu	or	writes	with	in	for	have	and	that
	it	be	this	is	not	are	but	of	to	the
CLASS 15	article	but	have	not	at	you	be	for	writes	on
	this	edu	it	of	and	the	in	to	that	is
CLASS 16	on	with	for	this	to	you	as	and	if	of
	that	but	are	is	the	have	in	be	not	it
CLASS 17	they	as	writes	not	if	on	have	are	be	of
	and	the	in	to	it	is	that	for	you	this
CLASS 18	article	have	be	on	this	as	writes	for	you	of
	and	is	by	not	that	are	the	in	to	it
CLASS 19	as	with	on	be	have	article	it	are	to	of
	is	in	you	writes	for	the	this	that	and	not
CLASS 20	article	but	with	on	be	that	and	are	the	of
	not	in	this	for	have	writes	it	you	is	to

Problem 3

Part a)

The problem of cost sensitive classification involves associating a certain cost penalty, c, in the case of a misclassification. In this case, we assign the cost of a "false positive" as \$$c$ and the cost of a "false negative" as \$1. We can express this cost penalty as follows:

$$\Pr(Y = 1|X) = c \cdot \Pr(Y = 0|X)$$

That is, the probability of classifying a new observation is $Y = 0$ is c times more likely than classifying it in $Y = 1$. Rearranging and using Bayes' Theorem, we can obtain the following expression:

$$c \cdot \Pr(Y = 0) \cdot \Pr(X|Y = 0) = \Pr(Y = 1) \cdot \Pr(X|Y = 1)$$

By substituting in the class priors and the p.d.f. of the normally distributed class conditionals, we can eventually simplify the expression and solve for x as a function of c.

$$\frac{2}{3} \cdot \frac{c}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} = \frac{1}{3} \cdot \frac{2}{\sqrt{2\pi}} e^{-2(x-1)^2}$$

$$c \cdot e^{-\frac{x^2}{2}} = e^{-2(x-1)^2}$$

$$\log(c) - \frac{x^2}{2} = -2(x-1)^2$$

$$2\log(c) - x^2 = -4x^2 + 8x - 4$$

Equating to zero and solving for x using the quadratic equation, we obtain:

$$x = \frac{8 \pm \sqrt{64 - 12 \cdot (4 + 2\log(c))}}{6}$$

$$x = \frac{8 \pm \sqrt{16 - 24\log(c)}}{6}$$

It is important to note that with $1 \leq c \leq 1.5$, the discriminant of the root will always remain positive and a finite decision boundary will exist. The general expression for the decision boundary is:

$$\left[\frac{8 - \sqrt{16 - 24\log(c)}}{6}, \frac{8 + \sqrt{16 - 24\log(c)}}{6}\right]$$

With $c = 1$ and $c = 1.5$ we obtain:

$$[0.6667, 2.000] \text{ and } [0.9160, 1.751]$$

Part b)

As we observed above, the decision boundary shrank in the second case and since $\sqrt{16 - 24\log(c)}$ is a strictly decreasing function, the decision boundary will eventually cease to exist as c increases. Solving for 0 in the discriminant, we obtain:

$$\sqrt{16 - 24\log(c)} = 0$$

$$\frac{16}{24} = \log(c)$$

$$c = 1.948$$

Now considering $c \geq 10$, we can see that there will be no decision region, and the classifier will always predict $Y = 0$ and $\Pr(Y = 0|X) = 1$.

Problem 4

Part a)

The probability of drawing balls of different classes is the probability of drawing a certain color on the first trial, then drawing any other color except the one you picked on the first trial next. Let:

$$A \coloneqq \{Drawing\ color\ c\}$$

Then it follows from independence that,

$$\Pr(A^c \cap A) = \Pr(A^c)\Pr(A) = (1 - \Pr(A))\Pr(A)$$

In general, the probability of drawing different colors when sampling with replacement is:

$$\left(1 - \frac{n_c}{100}\right)\left(\frac{n_c}{100}\right)$$

Where n_c represents the number of balls of color c.

Part b)

Painting the 100 balls uniformly (i.e. 20 balls for each color) maximizes the probability from part a). As stated earlier, the probability of drawing different colors when sampling with replacement is simply $(1 - \Pr(A))\Pr(A)$. Extending this to five classes, we obtain

$$\Pr(\text{different color}) = \sum_{i=1}^{5}(1 - \Pr(C_i))\Pr(C_i) \quad \text{for } C = (c_1, c_2, \dots, c_5) \in \mathcal{C}$$

$$\text{Where } \Pr(C_i) = \left(\frac{n_c}{100}\right)$$

$$\text{subject to: } \sum_{c=1}^{5} n_c = 1$$

To maximize this function, we can take the partial derivative for each of the five colors, equate it to zero and solve the system of equations:

$$\frac{\partial}{\partial n_i}\left(1-\left(\frac{n_i}{100}\right)\right)\left(\frac{n_i}{100}\right)=0 \quad \text{for each } i \in \{1,2,\ldots,5\}$$

We then obtain a system of four linear equations and four unknowns by using the constraint mentioned above: $n_5 = 1 - \sum_{i=1}^{4} n_i$. Then, solving for n_i, we obtain that a value of 20 for each color maximizes the system. To illustrate the point, we can use a concrete example:

$$\Pr(\text{different color}) = 5 \cdot \left(1-\frac{20}{100}\right)\left(\frac{20}{100}\right) = 0.8$$

If we modify the number of balls in a certain class, we can see the effect on the total probability:

$$\Pr(\text{different color}) = 3 \cdot \left(1-\frac{20}{100}\right)\left(\frac{20}{100}\right) + \left(1-\frac{19}{100}\right)\left(\frac{19}{100}\right) + \left(1-\frac{21}{100}\right)\left(\frac{21}{100}\right) = 0.7998$$

In fact, any deviation from the uniform case will result in a lower total probability. Thus, painting 20 balls for each color category maximizes the total probability.