CS6923 Machine Learning, Fall 2023
Prof. Sellie, NYU School of Engineering

Homework 2

You may work together with one other person on this homework. If you do that, *hand in JUST ONE homework for the two of you*, with both of your names on it. You may *discuss* this homework with other students but YOU MAY NOT SHARE WRITTEN ANSWERS OR CODE WITH ANYONE BUT YOUR PARTNER.

Part I: Written Exercises

1. Suppose there are two types of screenings that are used to detect whether a person has a certain disease. Screening method 1 is a blood test which is fast and inexpensive, but not very accurate. It has a 20% false positive rate, and a 15% false negative rate.

   Screening method 2 is based on doing an MRI of the patient. It is expensive and takes more time. It has a 7% false positive rate, and a 4% false negative rate.

   The "false positive rate" is the probability that the method mistakenly returns a positive result (says the person has the disease), when the person does not have the disease. The "false negative rate" is the probability that the method mistakenly returns a negative result (says the person doesn't have the disease), when the person *does* have the disease.

   Suppose that 0.01% of the population has this disease.

   (a) Suppose a random person from the population is screened using Screening method 1, and it returns a positive result. Using a Bayesian approach, which is the MAP hypothesis: that the person has the disease, or that the person does not have the disease?

   (b) Now suppose that because the person had a positive result using Screening method 1, the person is then sent to be screened using Screening method 2. If the result is positive again, what is the posterior probability (MAP) that the person has the disease? Assume the results of the two screening methods are independent.

2. To apply k-NN to problems with real-valued attributes, it is important to scale the attributes beforehand. One standard approach to scaling the values of an attribute x_i is to replace each x_i by the following value:

$$\frac{x_i - x_i^{min}}{x_i^{max} - x_i^{min}}$$

Here x_i^{min} and x_i^{max} are the minimum and maximum values of x_i in the training set. This ensures that all values of x_i in the training set will be between 0 and 1.

(a) Show the result of applying the above scaling formula to the following dataset (showing two decimal places for each). You should do the scaling SEPARATELY for each attribute (so you should compute a different min and max value for each attribute x_i). To get you started, we have already scaled x_1, so you only need to scale x_2.

Original Dataset

x1	x2	label
2.5	40	+
3.8	51	+
-0.3	-1	+
0.7	2	-
1.6	26	-
2.3	41	-

Scaled Dataset

x1	x2	label
0.68		+
1.00		+
0.00		+
0.24		-
0.46		-
0.63		-

(b) When applying scaling of the attributes in k-NN, it is important to scale any new (test) examples using the same formula that was used to scale the attributes in the training set. Using the scaling formula you computed for the training set above, scale the new example $x = \begin{bmatrix} 3.9 \\ 4 \end{bmatrix}$ and classify it using k-NN with the Euclidean distance measure, for $k = 1$. (The scaled version of the new example can have attribute values that are NOT between 0 and 1.)

What is the predicted label for the example? Show your work.

3. Consider the following small labeled dataset for a binary classification problem (classes + and -). There are two real-valued attributes, x_1 and x_2.

   (In practice, we wouldn't want to use a dataset this small to learn, but this is just for practice.)

x1	x2	label
2.7	4.8	+
3.2	5.1	+
-0.4	-0.3	+
0.6	0.5	-
1.8	2.8	-
2.1	4.3	-

   (a) Perform Quadradic Discriminanty Analysis (QDA) on the data above. (i.e. Fit one multivariate Gaussian to the data from the positive class, and another mulitivariate Gaussian to the data from the negative class. To classify a new example, predict using the MAP hypothesis.)

   (Since we have 2 variables, the two multivariate Gaussians are actually "bivariate" Gaussians. Sometimes people do not use matrix notation when describing the parameters of a bivariate Gaussian. However, we will use matrix notation here because it used for Gaussians in more than 2 dimensions, and we will also be using it later in the course.)

   In order to fit the two multivariate Gaussians, we need to compute estimates $\hat{\mu} = [\mu_1, \ldots, \mu_d]^T$ and $\hat{\mathbf{\Sigma}}$ of the mean and covariance matrix respectively, for each of the two distributions. (Note that the examples $\mathbf{x}^{(i)}$ are column vectors, and so is $\hat{\mu}$.)

   Calculate these estimates for the above dataset, using the ML estimates. The formulas for calculating the ML estimates of the mean and covariance matrix of a multivariate Gaussian, from a sample $\mathcal{X} = \{\mathbf{x}^{(t)}\}_{t=1}^{N}$, are as follows:

$$\hat{\mu} = \frac{\sum_{t=1}^{N} \mathbf{x}^{(t)}}{N}$$

$$\hat{\mathbf{\Sigma}} = \frac{\sum_{t=1}^{N} (\mathbf{x}^{(t)} - \hat{\mu})(\mathbf{x}^{(t)} - \hat{\mu})^T}{N}$$

   (Make sure to show which parameters belong to which distribution.)

   (b) Let $\mathbf{x} = \begin{pmatrix} 3.1 \\ 1.7 \end{pmatrix}$. Using QDA, predict the class of the example $\mathbf{x}$.

(c) Instead of using the approach above, which requires us to estimate separate covariance matrices for each class, we could assume that the covariance matrix for both classes is the same. Then, we could estimate a single covariance matrix using the entire dataset, rather than two covariances matrices. Give one reason it might be a good idea to do this, and one reason why it might not.

4. In this problem, you will use the Naive Bayes algorithm on a simple dataset, to predict the label of a new example.

   Consider the following small labeled dataset for a binary classification problem (classes + and -). There are three categorical attributes, x_1, x_2, and x_3. (A categorical attribute takes on values from a small, unordered set.) Attribute x_1 takes on values from the set $\{Medium, Low, High\}$, x_2 takes on values from the set $\{No, Yes\}$, and x_3 takes on values from the set $\{Red, Green\}$.

x1	x2	x3	label
Low	No	Red	+
Medium	No	Green	+
Low	No	Green	+
Low	Yes	Red	-
High	No	Green	-
Medium	Yes	Green	-
High	Yes	Green	-

   For example, suppose you were asked to estimate $P(x_1 = Medium|+)$, using add-m smoothing, for $m = 0.2$. Since x_1 has 3 possible values, there are 3 examples labeled +, and only 1 has $x_1 = Medium$, the estimate would be $\frac{1+0.2}{3+0.2*3}$.

   (a) Using add-m smoothing with $m = 0.2$, calculate estimates for $P(x_1 = Low|+)$, $P(x_2 = Yes|+)$, and $P(x_3 = Green|+)$. Then calculate estimates for $P(x_1 = Low|-)$, $P(x_2 = Yes|-)$, and $P(x_3 = Green|-)$.

   (b) Using the Naive Bayes assumption, if $x = [Low, Yes, Green]$ calculate (estimates of) $P(x|+)$ and $P(x|-)$, for $x = [Low, Yes, Green]$.

   (c) Based on the estimates of $P(x|+)$ and $P(x|-)$ that you just calculated, which is the ML label for the example $x = [Low, Yes, Green]$?

   (d) For the above dataset, an unsmoothed estimate of the priors $P(+)$ and $P(-)$, would be $P(+) = 3/7$ and $P(-) = 4/7$. Using these priors, and the values you calculated for $x = [Low, Yes, Green]$, what is the MAP label for the example $x = [Low, Yes, Green]$?

Part II: Programming and Questions

Complete the Python notebook to classify SMS messages as spam or not spam. In the Python notebook that was posted, the data set has been divided into training and test sets. (Real-world spam classification tasks *today* are much more difficult!)

The training example are in a Numpy array called X_train and their corresponding labels are in a Numpy array called y_train. The test examples are in a Numpy array called X_test and their corresponding labels are in a Numpy array called y_test.

To do: Implement Bernoulli Naive Bayes in Python.

For each test example $(x_1, \ldots, x_d)$, you will need to find the class maximizing

$$p(x_1|y) * p(x_2|y) * \ldots * p(x_d|y) * P(y)$$

using the estimates of the $p(x_j|y)$ and $p(y)$.

Training details: During the training phase, you will use the examples in the training file to estimate the value of $P(y)$ for each class ($y \in \{0, 1\}$). For each pair x_j, y of feature and class, you will need to estimate the probability $p(x_j|y)$.

Estimation of $P(y)$: To estimate the $P(y)$ values, just calculate the fraction of the training examples that are in class y.

Testing details: For each example $(x_1, \ldots, x_d)$ in the X_test and the labels in y_test, you want to determine which class maximizes

$$p(x_1|y) * p(x_2|y) * \ldots * p(x_d|y) * P(y)$$

Multiplying lots of small values can lead to underflow. To avoid that, you should not calculate

$$p(x_1|y) * p(x_2|y) * \ldots * p(x_d|y) * P(y)$$

directly for each class y. Instead, calculate $\log[p(x|y) * P(y)] = \log P(y) + \sum_{i=1}^{d} \log p(x_i|y)$. (Use the natural log, ln. But be sure to check to make sure the probability is not 0 before taking the log.) Then label the example with the class achieving the maximum value for this expression. If there is a tie, give the example the label 1.

Outputs

Your program should output the following values and write them to a separate text file (or the standard output) which you will NOT hand in.

- The estimated value of $P(y)$ for each class y.
- The estimates $\phi_{i|y}$ for the Bernoulli corresponding to $p(x_i|y)$, for each attribute x_i and each class y.
- The predicted classes for the first 50 test examples.
- Total number of test examples classified correctly. (You need to compare the predicted class for each test example with the given class label.)
- Total number of test examples classified incorrectly.
- The percentage error on the test examples.

Questions

Answers the following questions and put your answers in a pdf file called `proganswers.pdf`.

(a) What was the estimated value of $P(y)$ for $y = 1$?

(b) What was the estimated value of $P(y)$ for $y = 0$?

(c) What were the estimated values for $\phi_{admirer|y}$ for the corresponding feature `admirer` when $y = 1$ and for $y = 0$.

(d) What were the estimated values for $\phi_{secret|y}$ for the corresponding to feature `secret` when $y = 1$ and for $y = 0$.

(e) Which classes were predicted for the first 5 examples in the test set?

(f) Which classes were predicted for the last 5 examples in the test set?

(g) What was the percentage error on the examples in the test file?

(h) Repeat the above step (question 4g) by adding m smoothing by trying different values of m where $0 < m \leq 1$. Did the smoothing help? If so, for what value of m?

(i) Sometimes a not-very-intelligent learning algorithm can achieve high accuracy on a particular learning task simply because the task is easy. To check for this, you can compare the performance of your algorithm to the performance of some very simple algorithms. One such algorithm just predicts the majority class (the class that is most frequent in the training set). This algorithm is sometimes called Zero-R. It can achieve high accuracy in a 2-class problem if the dataset is very imbalanced (i.e., if the fraction of examples in one class is much larger than the fraction of examples in the other). What accuracy is attained is you use Zero-R instead of Bernoulli Naive Bayes?

If it is not obvious how to run your program, you should include a README file.