10-315 Machine Learning: Homework 2

Due 11:59 p.m. Friday, February 15, 2019

Instructions

- **Submit your homework on time electronically by submitting to Autolab.**.
- **Late homework policy**: Homework is worth full credit if submitted before the due date. Up to 50 % credit can be received if the submission is less than 48 hours late. The lowest homework grade at the end of the semester will be dropped. Please talk to the instructor in the case of extreme extenuating circumstances. **Note that, your scores on the coding part will also be penalized if you choose to submit the writeup after the due date.**
- **Collaboration policy**: You are welcome to collaborate on any of the questions with anybody you like. However, you must write up your own final solution, and you must list the names of anybody you collaborated with on this assignment.
- **Writeup Format:**
  - The written portion should be typeset (e.g. LaTeX).
  - The submission format should be PDF.
  - Every problem should be on a different page.

Problem 1: Independent events and Bayes Theorem

1. [**5 Points**] For events A, B prove:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B|A)P(A) + P(B|\neg A)P(\neg A)}.$$

($\neg A$ denote the event that A does not occur.)

2. Let X, Y, and Z be random variables taking values in $\{0, 1\}$. The following table lists the probability of each possible assignment of 0 and 1 to the variables X, Y, and Z:

	$Z = 0$		$Z = 1$	
	$X = 0$	$X = 1$	$X = 0$	$X = 1$
$Y = 0$	0.1	0.05	0.1	0.1
$Y = 1$	0.2	0.1	0.175	0.175

(a) [**4 Points**] Is X independent of Y? Why or why not?

(b) [**4 Points**] Is X conditionally independent of Y given Z? Why or why not?

(c) [**4 Points**] Calculate $P(X \neq Y|Z = 0)$.

Problem 2: Maximum Likelihood Estimation

This problem explores maximum likelihood estimation (MLE), which is a technique for estimating an unknown parameter of a probability distribution based on observed samples. Suppose we observe the values of n iid [1] random variables $X_1, \ldots, X_n$ drawn from a single Bernoulli distribution with parameter θ. In other words, for each X_i, we know that

$$P(X_i = 1) = \theta \quad \text{and} \quad P(X_i = 0) = 1 - \theta.$$

Our goal is to estimate the value of θ from these observed values of X_1 through X_n.

For any hypothetical value $\hat{\theta}$, we can compute the probability of observing the outcome $X_1, \ldots, X_n$ if the true parameter value θ were equal to $\hat{\theta}$. This probability of the observed data is often called the *data ikelihood*, and the function $L(\hat{\theta}) = P(X_1, \ldots, X_n|\hat{\theta})$ that maps each $\hat{\theta}$ to the corresponding likelihood is called the *likelihood function*. A natural way to estimate the unknown parameter θ is to choose the $\hat{\theta}$ that maximizes the likelihood function. Formally,

$$\hat{\theta}^{\text{MLE}} = \operatorname*{argmax}_{\hat{\theta}} L(\hat{\theta}).$$

Often it is more convenient to work with the log likelihood function $\ell(\hat{\theta}) = \log L(\hat{\theta})$. Since the log function is increasing, we also have

$$\hat{\theta}^{\text{MLE}} = \operatorname*{argmax}_{\hat{\theta}} \ell(\hat{\theta}).$$

1. [**6 Points**] Write a formula for the log likelihood function, $\ell(\hat{\theta})$. Your function should depend on the random variables $X_1, \ldots, X_n$, the hypothetical parameter $\hat{\theta}$, and should be simplified as far as possible (i.e., don't just write the definition of the log likelihood function). Does the log likelihood function depend on the order of the random variables?

[1] iid means Independent, Identically Distributed.

2. **[6 Points]** Consider the following sequence of 10 samples:

$$X = (0, 1, 0, 1, 1, 0, 0, 1, 1, 1).$$

Compute the maximum likelihood estimate for the 10 samples. Show all of your work (hint: recall that if x^* maximizes $f(x)$, then $f'(x^*) = 0$).

3. **[6 Points]** Now we will consider a related distribution. Suppose we observe the values of m iid random variables $Y_1, \ldots, Y_m$ drawn from a single Binomial distribution $B(n, \theta)$. A Binomial distribution models the number of 1's from a sequence of n independent Bernoulli variables with parameter θ. In other words,

$$P(Y_i = k) = \binom{n}{k} \theta^k (1-\theta)^{n-k} = \frac{n!}{k!(n-k)!} \cdot \theta^k (1-\theta)^{n-k}.$$

Write a formula for the log likelihood function, $\ell(\hat{\theta})$. Your function should depend on the random variables $Y_1, \ldots, Y_m$ and the hypothetical parameter $\hat{\theta}$.

4. **[6 Points]** Consider two Binomial random variables Y_1 and Y_2 with the same parameters, $n = 5$ and θ. The Bernoulli variables for Y_1 and Y_2 resulted in $(0, 1, 0, 1, 1)$ and $(0, 0, 1, 1, 1)$, respectively. Therefore, $Y_1 = 3$ and $Y_2 = 3$. Compute the maximum likelihood estimate for the 2 samples. Show your work.

5. **[4 Points]** How do your answers for parts 1 and 3 compare? What about parts 2 and 4? If you got the same or different answers, why was that the case?

Problem 3: Implementing Naive Bayes

In this question you will implement a Naive Bayes classifier for a text classification problem. You will be given a collection of text articles, each coming from either the serious European magazine *The Economist*, or from the not-so-serious American magazine *The Onion*. The goal is to learn a classifier that can distinguish between articles from each magazine.

We have pre-processed the articles so that they are easier to use in your experiments. We extracted the set of all words that occur in any of the articles. This set is called the *vocabulary* and we let V be the number of words in the vocabulary. For each article, we produced a feature vector $X = \langle X_1, \ldots, X_V \rangle$, where X_i is equal to 1 if the i^{th} word appears in the article and 0 otherwise. Each article is also accompanied by a class label of either 1 for The Economist or 2 for The Onion. Later in the question we give instructions for loading this data into Python.

When we apply the Naive Bayes classification algorithm, we make two assumptions about the data: first, we assume that our data is drawn iid from a joint probability distribution over the possible feature vectors X and the corresponding class labels Y; second, we assume for each pair of features X_i and X_j with $i \neq j$ that X_i is conditionally independent of X_j given the class label Y (this is the Naive Bayes assumption). Under these assumptions, a natural classification rule is as follows: Given a new input X, predict the most probable class label $\hat{Y}$ given X. Formally,

$$\hat{Y} = \underset{y}{\operatorname{argmax}}\, P(Y = y|X).$$

1. **[5 points]** Prove the classification rule can be rewritten as

$$\hat{Y} = \underset{y}{\operatorname{argmax}} \left(\prod_{w=1}^{V} P(X_w|Y = y) \right) P(Y = y).$$

2. **[5 points]** How many parameters are needed to represent the distribution $P(X|Y=y)$ when using the Naive Bayes assumption? How many are needed if we do not use the Naive Bayes assumption? Based on this difference, in which cases is there a big gain from making this assumption?

Of course, since we don't know the true joint distribution over feature vectors X and class labels Y, we need to estimate the probabilities $P(X|Y=y)$ and $P(Y=y)$ from the training data. For each word index $w \in \{1,\ldots,V\}$ and class label $y \in \{1,2\}$, the distribution of X_w given $Y=y$ is a Bernoulli distribution with parameter θ_{yw}. In other words, there is some unknown number θ_{yw} such that

$$P(X_w = 1|Y=y) = \theta_{yw} \quad \text{and} \quad P(X_w = 0|Y=y) = 1-\theta_{yw}.$$

For both The Economist and The Onion, we believe that each word w has a non-zero chance of appearing, but it is more likely that w will not occur in any particular document. We incorporate this belief by computing a MAP estimate using a Beta$(1.001, 1.9)$ prior on θ_{wy}. This has the added benefit of ensuring that none of our estimates of θ_{wy} are equal to 0 or 1 (which can cause problems for Naive Bayes).

Similarly, the distribution of Y (when we consider it alone) is a Bernoulli distribution (except taking values 1 and 2 instead of 0 and 1) with parameter ρ. In other words, there is some unknown number ρ such that

$$P(Y=1) = \rho \quad \text{and} \quad P(Y=2) = 1-\rho.$$

In this case, since we have many examples of articles from both The Economist and The Onion, there is no risk of having zero-probability estimates, so we will instead use the MLE.

Programming Instructions

Parts (3) through (5) of this question each ask you to implement one function related to the Naive Bayes classifier. You will submit your code online through the CMU autolab system by uploading `NB.py`, which will execute it remotely against a suite of tests. Your grade will be automatically determined from the testing results. Since you get immediate feedback after submitting your code and you are allowed to submit as many different versions as you like (without any penalty), it easy for you to check your code as you go.

Our autograder requires that you write your code using Python 3 and numpy 1.13.3. Otherwise, when running your program on Autolab, it may produce a result different from the result produced on your local computer

The file `hw2data.pkl` contains the data that you will use in this problem. You can load it into Python using `pickle`. After loading the data, you will see that there are 5 variables: `Vocabulary`, `XTrain`, `yTrain`, `XTest`, and `yTest`.

- `Vocabulary` is a $V \times 1$ dimensional array that that contains every word appearing in the documents. When we refer to the j^{th} word, we mean `Vocabulary[j,0]`.
- `XTrain` is a $n \times V$ dimensional matrix describing the n documents used for training your Naive Bayes classifier. The entry `XTrain[i,j]` is 1 if word j appears in the i^{th} training document and 0 otherwise.
- `yTrain` is a $n \times 1$ dimensional matrix containing the class labels for the training documents. `yTrain[i,0]` is 1 if the i^{th} document belongs to The Economist and 2 if it belongs to The Onion.

- Finally, XTest and yTest are the same as XTrain and yTrain, except instead of having n rows, they have m rows. This is the data you will test your classifier on and it should not be used for training.

Logspace Arithmetic

When working with very large or very small numbers (such as probabilities), it is useful to work in *logspace* to avoid numerical precision issues. In logspace, we keep track of the logs of numbers, instead of the numbers themselves. For example, if $p(x)$ and $p(y)$ are probability values, instead of storing $p(x)$ and $p(y)$ and computing $p(x) * p(y)$, we work in log space by storing $\log(p(x))$, $\log(p(y))$, and we can compute the log of the product, $\log(p(x) * p(y))$ by taking the sum: $\log(p(x) * p(y)) = log(p(x)) + log(p(y))$.

Training Naive Bayes

3. [**8 Points**] Complete the function `D = NB_XGivenY(XTrain, yTrain)`. The output D is a $2 \times V$ matrix, where for any word index $w \in \{1, \dots, V\}$ and class index $y \in \{1, 2\}$, the entry `D[y-1,w-1]` is the MAP estimate of $\theta_{yw} = P(X_w = 1|Y = y)$ with a Beta(1.001,1.9) prior distribution. To help with numerical issues clip D to be in $[10^{-5}, 1 - 10^{-5}]$ before this function returns it.

4. [**8 Points**] Complete the function `p = NB_YPrior(yTrain)`. The output p is the MLE for $\rho = P(Y \ = \ 1)$.

5. [**8 Points**] Complete the function `yHat = NB_Classify(D, p, X)`. The input X is an $m \times V$ matrix containing m feature vectors (stored as its rows). The output yHat is a $m \times 1$ vector of predicted class labels, where `yHat[i]` is the predicted label for the i^{th} row of X. [Hint: In this function, you will want to use Logspace Arithmetic function to avoid numerical problems.]

Questions

6. [**5 Points**] Train your classifier on the data contained in XTrain and yTrain by running

```
D = NB_XGivenY(XTrain, yTrain)
p = NB_YPrior(yTrain)
```

Use the learned classifier to predict the labels for the article feature vectors in XTrain and XTest by running

```
yHatTrain = NB_Classify(D, p, XTrain)
yHatTest = NB_Classify(D, p, XTest)
```

Use the function ClassificationError to measure and report the training and testing error by running

```
trainError = ClassificationError(yHatTrain, yTrain)
testError = ClassificationError(yHatTest, yTest)
```

How do the train and test errors compare? Which is more representative of the error we would expect to have on a new collection of articles? Does Naive Bayes attempt to minimize the training error?

7. [**8 Points**] In this question we explore how the size of the training data set affects the test and train error. For each value of m in $\{100, 130, 160, \ldots, 580\}$, train your Naive Bayes classifier on the first m training examples (that is, use the data given by `XTrain[0:m,]` and `yTrain[0:m]`). Plot the training and testing error for each such value of m. The x-axis of your plot should be m, the y-axis should be error, and there should be one curve for training error and one curve for testing error. Explain the general trend of both the training and testing error curves.

8. [**8 Points**] Finally, we will try to interpret the learned parameters. Train your classifier on the data contained in `XTrain` and `yTrain`. For each class label $y \in \{1, 2\}$, create three lists according to the following criteria (Note that some of the words may look a little strange because we have run them through a stemming algorithm that tries to make words with common roots look the same. For example, "stemming" and "stemmed" would both become "stem"):

   - Top five words that the model says are most likely to occur in a document from class y. That is, the top five words according to this metric:

   $$P(X_w = 1|Y = y)$$

   - Top five words w according to this metric:

   $$\frac{P(X_w = 1|Y = y)}{P(X_w = 1|Y \neq y)}.$$

   Which list of words is more informative about the class y? Briefly explain your reasoning.