CS440/ECE448 Fall 2016

Assignment 3: Naive Bayes Classification

Due date: Monday, November 14, 11:59:59 PM

The goal of this assignment is to implement Naive Bayes classifiers as described in this lecture and to apply it to the task of classifying visual patterns and text documents. As before, you can work in teams of up to three people (three-credit students with three-credit students, four-credit students with four-credit students).

Contents

- Part 1: Digit classification
  - Part 1.1 (for everybody): Single pixels as features
  - Part 1.2 (for four-credit students): Pixel groups as features
- Part 2: Text document classification
  - Part 2.1 (for everybody): Sentiment analysis of movie reviews and binary conversation topic labeling
  - Part 2.2 (for four-credit students): Conversation topic labels identification from a 40-topic dataset
- Report checklist
- Submission instructions

Part 1: Digit classification
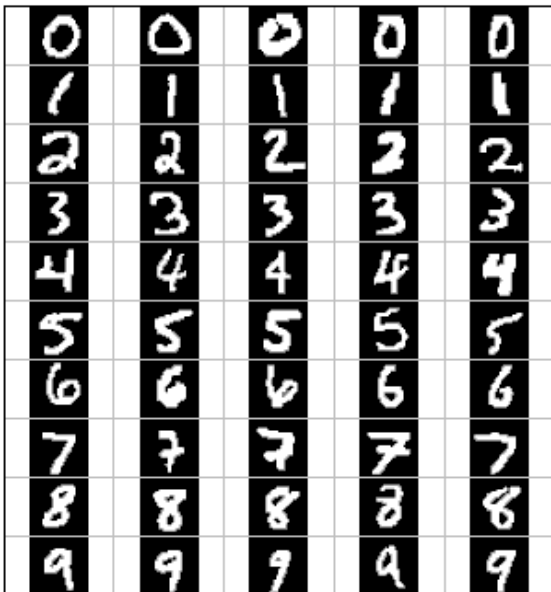
(Adapted from Berkeley CS 188 project 5)

Data: This file is a zip archive containing training and test digits, together with their ground truth labels (see `readme.txt` in the zip archive for an explanation of the data format). There are `5000` training exemplars (roughly `500` per class) and `1000` test exemplars (roughly `100` per class).

Part 1.1 (for everybody): Single pixels as features

- **Features:** The basic feature set consists of a single binary indicator feature for each pixel. Specifically, the feature F_{ij} indicates the status of the (i,j)-th pixel. Its value is `1` if the pixel is foreground (no need to distinguish between the two different foreground values), and `0` if it is background. The images are of size `28*28`, so there are `784` features in total.

- **Training:** The goal of the training stage is to estimate the **likelihoods** `P(Fij | class)` for every pixel location `(i,j)` and for every digit class from `0` to `9`. The likelihood estimate is defined as

  `P(Fij = f | class) = (# of times pixel (i,j) has value f in training examples from this class) / (Total # of training examples from this class)`

  In addition, as discussed in the lecture, you have to **smooth** the likelihoods to ensure that there are no zero counts. *Laplace smoothing* is a very simple method that increases the observation count of every value `f` by some constant `k`. This corresponds to adding `k` to the numerator above, and `k*V` to the denominator (where V is the number of possible values the feature can take on). The higher the value of k, the stronger the smoothing. Experiment with different values of k (say, from 0.1 to 10) and find the one that gives the highest classification accuracy.

  You should also estimate the **priors** `P(class)` by the empirical frequencies of different classes in the training set.

- **Testing:** You will perform **maximum a posteriori (MAP)** classification of test digits according to the learned Naive Bayes model. Suppose a test image has feature values $f_{1,1}$, $f_{1,2}$, ... , $f_{28,28}$. According to this model, the posterior probability (up to scale) of each class given the digit is given by

  `P(class) · P(f`$_{1,1}$` | class) · P(f`$_{1,2}$` | class) · ... · P(f`$_{28,28}$` | class)`

  Note that in order to avoid underflow, it is standard to work with the log of the above quantity:

  `log P(class) + log P(f`$_{1,1}$` | class) + log P(f`$_{1,2}$` | class) + ... + log P(f`$_{28,28}$` | class)`

  After you compute the above decision function values for all ten classes for every test image, you will use them for MAP classification.

- **Evaluation:** Use the true class labels of the test images from the `testlabels` file to check the correctness of the estimated label for each test digit. Report your performance in terms of the **classification rate for each digit** (percentage of all test images of a given digit correctly classified). Also report your **confusion matrix**. This is a 10x10 matrix whose entry in row r and column c is the percentage of test images from class r that are classified as class c. In addition, for each digit class, show the test examples from that class that have the highest and the lowest posterior probabilities according to your classifier. You can think of these as the most and least "prototypical" instances of each digit class (and the least "prototypical" one is probably misclassified).

  Important: The ground truth labels of test images should be used *only* to evaluate classification accuracy. They should not be used in any way during the decision process.

  Tip: You should be able to achieve at least 70% accuracy on the test set. One "warning sign" that you have a bug in your implementation is if some digit gets 100% or 0% classification accuracy (that is, your system either labels all the test images as the same class, or never wants to label any test images as some particular class).

- **Odds ratios:** When using classifiers in real domains, it is important to be able to inspect what they have learned. One way to inspect a naive Bayes model is to look at the most likely features for a given label. Another tool for understanding the parameters is to look at odds ratios. For each pixel feature `Fij` and pair of classes `c1`, `c2`, the odds ratio is defined as

  `odds(Fij=1, c1, c2) = P(Fij=1 | c1) / P(Fij=1 | c2).`

  This ratio will be greater than one for features which cause belief in c_1 to increase over the belief in c_2. The features that have the greatest impact on classification are those with both a high probability (because they appear often in the data) and a high odds ratio (because they strongly bias one label versus another).

  Take four pairs of digits that have the highest confusion rates according to your confusion matrix, and for each pair, display the maps of feature likelihoods for both classes as well as the odds ratio for the two classes. For example, the figure below shows the log likelihood maps for 1 (left), 8 (center), and the log odds ratio for 1 over 8 (right):
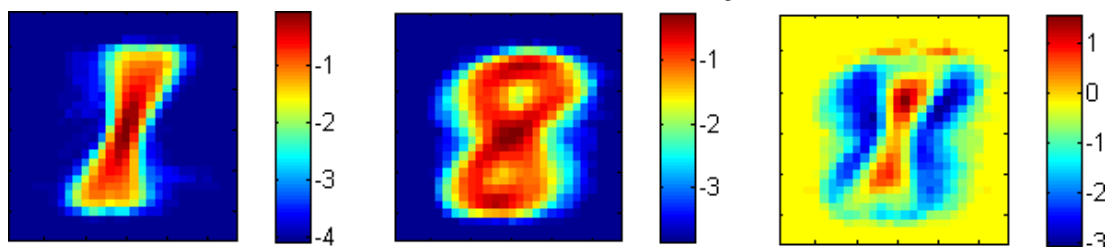
If you cannot do a graphical display like the one above, you can display the maps in ASCII format using some coding scheme of your choice. For example, for the odds ratio map, you can use '+' to denote features with positive log odds, ' ' for features with log odds close to 1, and '-' for features with negative log odds.

Part 1.2 (for four-credit students): Pixel groups as features

Credit: Yanglei Song

Instead of each feature corresponding to a single pixel, we can form features from groups of adjacent pixels. We can view this as a relaxation of the Naive Bayes assumption that allows us to have a more accurate model of the dependencies between the individual random variables. Specifically, consider a 2*2 square of pixels with top left coordinate i,j and define a feature $G_{i,j}$ that corresponds to the ordered tuple of the four pixel values. For example, in the figure below, we have

$$G_{1,1} = (F_{1,1}, F_{1,2}, F_{2,1}, F_{2,2}).$$
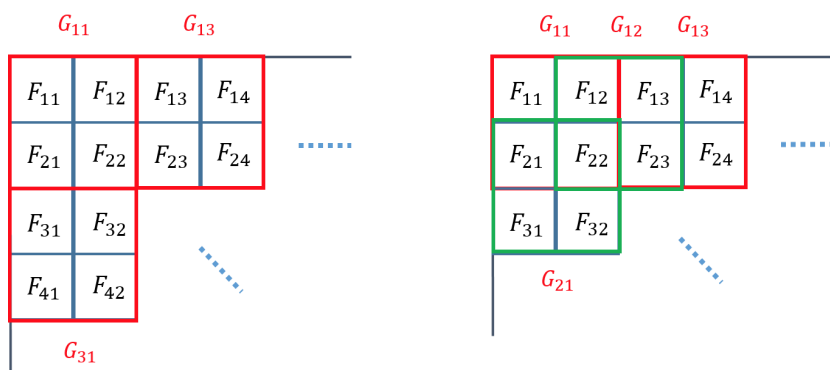
(The exact ordering of the four pixel values is not important as long as it's consistent throughout your implementation.) Clearly, this feature can have 16 discrete values. The 2*2 squares can be disjoint (left side of figure) or overlapping (right side of figure). In the case of disjoint squares, there are 14 * 14 = 196 features; in the case of overlapping squares, there are 27 * 27 = 729 features.

We can generalize the above examples of 2*2 features to define features corresponding to n*m disjoint or overlapping pixel patches. An n*m feature will have 2^{n*m} distinct values, and as many entries in the conditional probability table for each class. Laplace smoothing applies to these features analogously as to the single pixel features.

In this part, you should build Naive Bayes classifiers for feature sets of n*m disjoint/overlapping pixel patches and report the following:

- Test set accuracies for disjoint patches of size 2*2, 2*4, 4*2, 4*4.
- Test set accuracies for overlapping patches of size 2*2, 2*4, 4*2, 4*4, 2*3, 3*2, 3*3.
- Discussion of the trends you have observed for the different feature sets (including single pixels), in particular, why certain features work better than others for this task.
- Brief discussion of running time for training and testing for the different feature sets (which ones are faster and why, and how does the running time scale with feature set size).

Tip: You should be able to achieve over 80% accuracy with your best feature set.

Part 1 Extra Credit

- Experiment with yet more features to improve the accuracy of the Naive Bayes model. For example, instead of using binary pixel values, implement ternary features.

- Apply your Naive Bayes classifier with various features to this [face data](). It is in a similar format to that of the digit data, and contains training and test images and binary labels, where 0 corresponds to 'non-face' and 1 corresponds to 'face'. The images themselves are higher-resolution than the digit images, and each pixel value is either '#', corresponding to an edge being found at that location, or ' ', corresponding to a non-edge pixel.

Part 2: Text Document Classification

Created by Xuesong Yang based on the corpora of [Fisher English Transcripts Part 1](), [Part 2](), and [Movie Review Data]().

Part 2.1: For everybody

The goal of this part aims to apply Naive Bayes classifiers to solve basic binary classification problems in the area of natural language processing, e.g., sentiment analysis of movie reviews, conversation topic identification. You need to implement two models on two separate corpora as described below.

Models: For each dataset, you will build two different Naive Bayes classifiers. Parameters of these two models should be estimated on the training data, and the class labels of the test documents should be predicted based on your trained models. The two Naive Bayes models are defined as follows:

- **Multinomial Naive Bayes:** This is the model described in [the lecture](), where a document of length N has variables $W_1, \ldots, W_N$ and each variable W_i takes on values from 1 to V, where V is the size of the vocabulary. You can estimate the likelihoods $P(W_i = k \mid class)$ by frequency counts, as explained in the lecture.

- **Bernoulli Naive Bayes:** In this model, every document is described by V binary variables $W_1, \ldots, W_V$, and $W_i = 1$ if the i-th word appears *at least once* in the document, and 0 otherwise. You can estimate the likelihoods $P(W_i = 1 \mid class)$ as the proportion of documents from that class that feature the i-th word.

Datasets: You will apply your models to the two following corpora:

- **Sentiment analysis of movie reviews:** This corpus has 4000 training documents (2000 positive and 2000 negative), and 1000 test documents (500 positive and 500 negative), where a label of -1 denotes a negative review, and 1 denotes a positive review.

- **Binary conversation topic classification:** This corpus is generated from transcripts of two-person telephone conversations. Before the start of each conversation, two participants are assigned with a specific topic, for example, "Minimum Wage", "Reality TV Shows", "Pets", etc. In this binary classification task, the corpus has been filtered so that it only contains conversations on two topics: "Minimum Wage" and "Life Partners". The corpus has 878 training conversations (440 for Minimum Wage, 438 for Life Partners), and 98 test conversations (49 for Minimum Wage, 49 for Life Partners). For the target labels, -1 denotes Life Partners, and 1 denotes a topic relevant to Minimum Wage.

Train and test data are both preprocessed. Words that carry little meaning, like "the" and "and", have been removed (these are called "stop words"); the remaining words are sorted, counted, and presented in the following format:

```
label word_1:count_1 word_2:count_2 ... word_n:count_n
```

where `label` denotes the target topic label, `word_n:count_n` pair denotes the form of a word, and the number of times that it occurred in the document. The `word:count` features are each separated by a single space.

[Implementation Hints]

1. Create dictionaries consisting of all unique words occurring in the training documents.

2. Estimate conditional probability tables over these dictionaries for each class. Be sure to use Laplace smoothing. Note that there will be words in the test documents that do not occur in the dictionary; simply ignore those.

3. Conditional probability tables may not be the only parameters to estimate. To obtain better performance, you may also want to estimate the prior distribution of class labels.

Report: For each of the two models on two datasets, please report the following:

- The overall accuracy and confusion matrix as your model performance on the test data. You should be able to achieve around `90%` accuracy on the topical theme classification task for both models, and around `70%` accuracy on the sentiment analysis of movie review task.
- The top 10 words with the highest *likelihood* for each class, and the top 10 words with the highest *odds ratio* (are these different and why?).

Part 2.2: For four-credit students

Besides the experiments in Part 2.1, you should perform conversation topic classification on the full 40-topic corpus. This data contains 40 topic themes as shown in topic_description, and has 10244 training conversations and 1154 test conversations. The histograms of class labels on training data is shown below.

Report: For each of the two models, report the following:

- Overall accuracy score and confusion matrix of your Multinomial and Bernoulli classifiers. Which one is more accurate and why?
- For your more accurate classifier, take each topic and say with what other topic it was most commonly confused.

(10244 conversations, 40 topics)
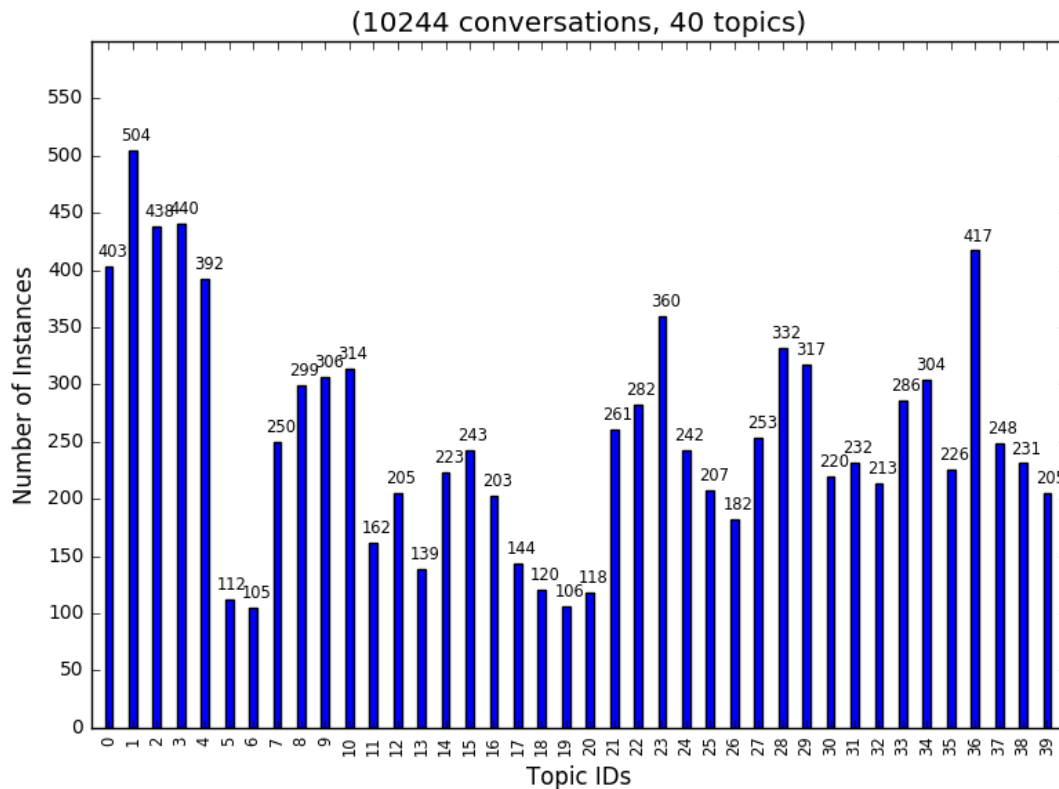
Extra Credit for Part 2

- Experiment with advanced techniques for improving performance of Naive Bayes on this dataset, such as lemmatization and tf-idf weighting (not covered in class).
- Experiments using other distribution for conditional probability distribution assumption.
- Visualize the bag-of-words representations of the documents using word cloud maps.
- Visualize the confusion matrix of your results using heatmaps for example.

Report Checklist

Part 1:

1. For everybody:

   - Briefly discuss your implementation, especially the choice of the smoothing constant.
   - Report classification rate for each digit and confusion matrix.
   - For each digit, show the test examples from that class that have the highest and lowest posterior probabilities according to your classifier.
   - Take four pairs of digits that have the highest confusion rates, and for each pair, display feature likelihoods and odds ratio.

2. For four-credit students:

   - Report test set accuracies for disjoint patches of size 2*2, 2*4, 4*2, 4*4, and for overlapping patches of size 2*2, 2*4, 4*2, 4*4, 2*3, 3*2, 3*3.
   - Discuss trends for the different feature sets.
   - Discuss training and testing running time for different feature sets.

Part 2:

1. For everybody:

   - For Multinomial and Bernoulli models, and for both datasets, report the classification rate for each class and show the confusi likelihood and the top 10 words with the highest odds ratio (discuss differences).

2. For four-credit students:

   - Give overall accuracy and confusion matrix for both classifiers, discuss differences in accuracy.
   - For the more accurate classifier, for each topic, say with what other topic it was most commonly confused.

Extra credit:

- We reserve the right to give **bonus points** for any advanced exploration or especially challenging or creative solutions that you implement. Three-credit students always get extra credit for submitting solutions to four-credit problems (at 50% discount). **If you submit any work for bonus points, be sure it is clearly indicated in your report.**

Statement of individual contribution:

- All group reports need to include a brief summary of which group member was responsible for which parts of the solution and submitted material. We reserve the right to contact group members individually to verify this information.

WARNING: You will not get credit for any solutions that you have obtained, but not included in your report! For example, if your code prints out path cost and number of nodes expanded on each input, but you do not put down the actual numbers in your report, or if you include pictures/files of your output solutions in the zip file but not in your PDF. The only exception is animated paths (videos or animated gifs).

Submission Instructions

As before, **one designated person from the group** will need to submit on **Compass 2g** by the deadline. Three-credit students must upload under **Assignment 3 (three credits)** and four-credit students must upload under **Assignment 3 (four credits)**. Each submission must consist of the following two attachments:

1. A **report** in **PDF format**. As before, the report should briefly describe your implemented solution and fully answer all the questions posed above. **Remember: you will not get credit for any solutions you have obtained, but not**

included in the report.

All group reports need to include a brief **statement of individual contribution**, i.e., which group member was responsible for which parts of the solution and submitted material.

The name of the report file should be **lastname_firstname_assignment3.pdf**. Don't forget to include the names of all group members and the number of credit credits at the top of the report.

2. Your **source code** compressed to a **single ZIP file**. The code should be well commented, and it should be easy to see the correspondence between what's in the code and what's in the report. You don't need to include executables or various supporting files (e.g., utility libraries) whose content is irrelevant to the assignment. If we find it necessary to run your code in order to evaluate your solution, we will get in touch with you.

The name of the code archive should be **lastname_firstname_assignment3.zip**.

Multiple attempts will be allowed but in most circumstances, only the last submission will be graded. **We reserve the right to take off points for not following directions.**

Late policy: For every day that your assignment is late, your score gets multiplied by 0.75. The penalty gets saturated after four days, that is, you can still get up to about 32% of the original points by turning in the assignment at all. If you have a compelling reason for not being able to submit the assignment on time and would like to make a special arrangement, you must send me email **at least a week before the due date** (any genuine emergency situations will be handled on an individual basis).

Be sure to also refer to **course policies** on academic integrity, etc.