Document Classifier s

In this project, you will implement the Naive Bayes and Logistic Regression document classifier and apply it to the classic 20 newsgroups dataset. In this dataset, each document is a posting that was made to one of 20 different usenet newsgroups.

Your goal is to write a program which can predict which newsgroup a given document was posted to.

Model 1 (do not implement this)

Say we have a document D containing d words; call the words $\{X_1, ..., X_d\}$. The value of random variable X_i is the word found in position i in the document. We wish to predict the label Y of the document, which can be one of m categories, We could use the model:

$$P(Y|X_1...X_d) \propto P(X_1...X_d|Y)P(Y) = P(Y)\prod P(X_i|Y)$$

That is, each X_i is sampled from some distribution that depends on its position X_i and the document category Y. As usual with discrete data, we assume that $P(X_i|Y)$ is a multinomial distribution over some vocabulary V; that is, each X_i can take one of $|V|$ possible values corresponding to the words in the vocabulary. Therefore, in this model, we are assuming (roughly) that for any pair of document positions i and j, $P(X_i|Y)$ may be completely different from $P(X_j|Y)$.

Question 1: In your answer sheet, explain in a sentence or two why it would be difficult to accurately estimate the parameters of this model on a reasonable set of documents (e.g. 1000 documents, each 1000 words long, where each word comes from a 50,000 word vocabulary). **[5 points]**

Model 2 (implement this)

To improve the model, make the additional assumption that: $\forall i, j \quad P(X_i|Y) = p(X_j|Y)$

Thus, in addition to estimating $P(Y)$, you must estimate the parameters for the single distribution $P(X|Y)$, which we define to be equal to $P(X_i|Y)$ for all X_i. Each word in a document is assumed to be an iid draw from this distribution.

Implementation

Your first task is to implement the Naive Bayes classifier specified above. You should estimate $P(Y)$ using the MLE, and estimate $P(X|Y)$ using a MAP estimate with the prior distribution Dirichlet$(1+\beta, ..., 1+\beta)$, where $\beta = 1/|V|$ and V is vocabulary.

MLE for P(Y)

$$P(Y_k) = \frac{\#\ of\ docs\ labeled\ Y_k}{total\ \#\ of\ docs}$$

MAP for P(X|Y)

$$P(X_i|Y_k) = \frac{(count\ of\ X_i\ in\ Y_k)+(\alpha-1)}{(total\ words\ in Y_k)+((\alpha-1)*(length\ of\ vocab\ list))}$$

where $\alpha = 1+\beta, \beta = \frac{1}{|V|}$

Classify

$$Y^{new} = argmax\left[\ log_2(P(Y_k)) + \sum_i (\#\ of\ X_i^{new}) log_2(P(X_i|Y_k))\right]$$

Priors and Overfitting

In your initial implementation, you used a prior Dirichlet(1 + β, ..., 1 + β) to estimate P (X|Y), and you set β = 1/|V |. In practice, the choice of prior_is a difficult question in Bayesian learning: either we must use domain knowledge, or we must look at the performance of different values on some validation set. Here we will use the performance on the testing set to gauge the effect of β

Question 2: Re-train your Naive Bayes classifier for values of β between .00001 and 1 and report the accuracy over the test set for each value of β. Create a plot with values of β on the x-axis and accuracy on the y-axis. Use a logarithmic scale for the x-axis (in Matlab, the semilogx command). Explain in a few sentences why accuracy drops for both small and large values of β **[5 points]**

Model 3 (implement this too)

Logistic Regression maximizes the conditional data likelihood as follows:

$$\ln P(\mathcal{D}_Y \mid \mathcal{D}_{\mathbf{X}}, \mathbf{w}) = \sum_{j=1}^{N} \ln P(y^j \mid \mathbf{x}^j, \mathbf{w})$$

$$= \sum_j y^j (w_0 + \sum_i^n w_i x_i^j) - \ln(1 + exp(w_0 + \sum_i^n w_i x_i^j))$$

This function is concave and can be optimized using the gradient ascent/descent algorithm.
For all the background in Logistic Regression read Tom Mitchell's online chapter: **http://www.cs.cmu.edu/~tom/mlbook/NBayesLogReg.pdf. Note especially section 3.**

Implementation

Implement a multinomial Logistic Regression and Gradient descent

Assuming row-major order, so the first index refers to a row, and the second refers to a column.

Given: m, the number of examples
k, the number of classes
n, the number of attributes each example has
η, a learning rate
λ, a penalty term
Δ, a $k \times m$ matrix where $\Delta_{ji} = \delta(Y^i = y_j)$ (using the delta equation as found in equation (29) in the Mitchell chapter)
X, an $m \times (n+1)$ matrix of examples, where $\forall i, X_{i0} = 1$, and X_{i1} through X_{in} are the attributes for example i
Y, an $m \times 1$ vector of true classifications for each example
W, a $k \times (n+1)$ matrix of weights
$P(Y|W, X) \sim exp(WX^T)$, a $k \times m$ matrix of probability values. To follow the format of equations (27) and (28) in the text, fill the last row with all 1's, and then normalize each column to sum to one by dividing the each value in the column by the sum of the column.

Then the update step for the logistic regression is

$$W^{t+1} = W^t + \eta((\Delta - P(Y|W, X))X - \lambda W^t)$$

Question 3: Re-train your Logistic Regression classifier for values of η starting from 0.01 to 0.001, λ = 0.01 to 0.001 and vary your stopping criterium from number of total iterations (e.g. 10,000) or = 0.00001 and report the accuracy over the test set for each value (this is more efficient if you plot your parameter values vs

accuracy) . Explain in a few sentences your observations with respect to accuracies and sweet spots [**5 points**]

Analysis of Results

Question 4: In your answer sheet, report your overall testing accuracy (Number of correctly classified documents in the test set over the total number of test documents), and print out the confusion matrix (the matrix C, where c_{ij} is the number of times a document with ground truth category j was classified as category i). **[5 points]**

Question 5: Are there any newsgroups that the algorithm(s) confuse more often than others? Why do you think this is? **[5 points]**

Identifying Important Features

One useful property of Naive Bayes is that its simplicity makes it easy to understand why the classifier behaves the way it does. This can be useful both while debugging your algorithm and for understanding your dataset in general. For example, it is possible to identify which words are strong indicators of the category labels we're interested in.

Question 6: Propose a method for ranking the words in the dataset based on how much the classifier 'relies on' them when performing its classification (hint: information theory will help). Your metric should use only the classifier's estimates of P (Y) and P (X|Y). It should give high scores to those words that appear frequently in one or a few of the newsgroups but not in other ones. Words that are used frequently in general English ('the', 'of', etc.) should have lower scores, as well as words that only appear extremely rarely throughout the whole dataset. Finally, in your method this should be an overall ranking for the words, not a per-category ranking**.[5 points]**

Question 7: Implement your method, set β back to $1/|V|$, and print out the 100 words with the highest measure. **[5 points]**

Question 8: If the points in the training dataset were not sampled independently at random from the same distribution of data we plan to classify in the future, we might call that training set biased. Dataset bias is a problem because the performance of a classifier on a biased dataset will not accurately reflect its future performance in the real world. Look again at the words your classifier is 'relying on'. Do you see any signs of dataset bias? **[5 points]**

Turn in the following:

Your code. Submit your file through UNM Learn. **The due date is Midnight (+ 3 hrs buffer) contingent to the late policy stated in the syllabus**. Your code should contain appropriate comments to facilitate understanding. If needed, your code must contain a Makefile or an executable script that receives the paths to the training and testing files

A report of about 6 to 12 pages that includes:

- A high-level description on how your code works.
- The accuracies you obtain under various settings.
- Explain which options work well and why.
- Answers to questions 1 to 8

Rubric:

Your code is thoroughly commented (10 pts)
You provided a well documented README file (10 pts)
Implementation of Naïve Bayes is correct (15 pts)
Implementation of Logistic Regression is correct (15 pts)
Your report is clear, concise and well organized (10 pts)
Your answers to questions 1 to 7 (40 pts)

TOTAL: 100 pts (10 pts of your final grade)