

# CS114 (Spring 2018) Homework 3

## Naive Bayes Classifier and Evaluation

Due March 2, 2018

You are given `naive_bayes.py`, and `movie_reviews.zip`, the NLTK movie review corpus. Reviews are separated into a training set (80% of the data) and a development set (10% of the data). A testing set (10% of the data) has been held out and is not given to you. Within each set, reviews are sorted by sentiment (positive/negative). The files are already tokenized. Each review is in its own file.

### Assignment

Your task is to implement a multinomial Naive Bayes classifier using bag-of-words features. Specifically, in `naive_bayes.py`, you should fill in the following functions:

- `train(self, train_set)`: This function should, given a folder of training documents, fill in `self.prior` and `self.likelihood`. The two dictionaries should satisfy the following conditions:
  - `self.prior[class] = log(P(class))`
  - `self.likelihood[feature][class] = log(P(feature|class))`

As your initial feature set, you should use the words ‘great’, ‘poor’, and ‘long’. Also, be sure to use add-one smoothing.

- `test(self, dev_set)`: This function should, given a folder of development (or testing) documents, return a dictionary of **results** such that:

- `results[filename]['correct']` = correct class
- `results[filename]['predicted']` = predicted class

Specifically, for each document, you should calculate the predicted class  $c_{NB}$  as follows (where  $C$  is your set of classes, positions refers to the positions in the document, and  $x_i$  is the word at position  $i$ ):

$$c_{NB} = \arg \max_{c_j \in C} (\log(P(c_j)) - \sum_{i \in \text{positions}} \log(P(x_i|c_j)))$$

- `evaluate(self, results)`: This function should, given the results of `test`, compute precision, recall, and F1 score for each class, as well as the overall accuracy. Recall that (where  $c_{ij}$  is the number of documents actually in class  $i$  that were classified as being in class  $j$ ):

- precision =  $\frac{c_{ii}}{\sum_j c_{ji}}$
- recall =  $\frac{c_{ii}}{\sum_j c_{ij}}$
- F1 =  $\frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$
- accuracy =  $\frac{\sum_i c_{ii}}{\sum_j \sum_i c_{ij}}$

Congratulations, you have implemented your very own Naive Bayes classifier! At this point, you should experiment with additional features. Perhaps the word ‘good’ would be a good feature to distinguish good reviews from bad ones. On the other hand, the word ‘long’ might not be a good feature to include. You should try at least seven more features (for a total of ten). Turn in your model that performs best on the development set.

## Tips and Tricks

1. You may find it helpful to look at your code from Homework 2, especially when collecting counts of features or performing add-one smoothing.
2. Python does not have a built-in `argmax` function, but it can be approximated using `max` and `itemgetter`. Specifically, for some list `lst` of tuples `(x, y)`, the value of `x` with the maximum `y` is:  
`max(lst, itemgetter(1)) [0]`.
3. When calculating your evaluation metrics, it may be helpful to create a confusion matrix. The `confusion_matrix` defined in `evaluate` can be populated as follows: `confusion_matrix[class_1][class_2]` = the number of documents actually in `class_1` that were classified as being in `class_2`.

## Write-up

You should also prepare a (very short!) write-up that includes the following (which will count toward your grade):

- What additional features you included/tried in your classifier
- Your evaluation results on the development set

Please also include the following (which will *not* count toward your grade):

- (About) how many hours you spent working on this assignment
- Any parts of the assignment you found particularly easy or difficult
- Any other comments on the assignment you would like to make

## Submission Instructions

Please submit two files: your write-up (PDF or plain-text format, please!), and `naive_bayes.py`. Do not include `movie_reviews.zip`.