Calvin Wang, 9501313, W19, CS165A

MP1: Semantic Analysis with BoW, TF-IDF, Gaussian Naive Bayes, and Multinomial Naive Bayes

Introduction

Throughout the training process, I decided to assign the negative sentiment class as 0 and the positive sentiment class as 1. The goal here is to be able to take a review and classify the sentiment as positive or negative. To speed up data processing, I took advantage of the python yield method so that my RAM wouldn't be overloaded with a big dataset at once.

Feature Extracting

The goal of feature extraction is to take an arbitrary of data and map it to a float value. This makes it more manageable for training classifiers. The two feature extraction methods attempted are Bag of Words and TF-IDF.

When building the Bag of Words (BoW) feature extraction method, I used a dictionary to keep track of the occurrence off each word throughout the positive or negative dataset of reviews. Once the dictionary was created, I would iterate through each word and add them into their respected feature with the BoW coefficient from the formula. I included Laplace Smoothing in this step with +1 in the numerator and setting the denominator as the sum of total number of words in the respected dataset and the total number of words in both datasets.

The same approach was taken for the Term Frequency Inverse Document Frequency (TF-IDF) feature extraction. I had TF be the count of each word and divided it by the total number of words in the respected training dataset. And then IDF be the log of the number of reviews over the number of reviews the word appeared in.

Training Classifiers

The goal of training classifiers is to build a pipeline that accepts novel reviews and classifies a sentiment be it positive or negative. Two variants of Naive Bayes were applied here: Gaussian and Multinomial

The Gaussian Naive Bayes (GNB) method utilized a Gaussian distribution to model the respected feature extracted from the dataset — be it BoW or TF-IDF. The approach was easily modularized given both feature extractors resulted in a direct mapping of word to floating point value. In terms of GNB, there are two parameters (mean and variance) that we needed to train for each class, resulting in four total parameters. Once the features were extracted through BoW or TF-IDF, I found the mean and standard deviation for each class and stored them in my weights. When it came time for testing, I tested each word of the test review into this equation:

$$f(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

In this equation, x is each word from the test review, mu and sigma^2 are w.r.t to the class being tested. Essentially, each word for each test review is run twice through this equation. And once the final probabilties were calculated, I compared the values of the two and took the max as the classification label.

The Multinomial Naive Bayes (MNB) method utilized a similar approach but with a Multinomial distribution as opposed to the Gaussian. In this case, I took the direct feature value extracted for the respected word and summed up the values across the test review.

Additional Smoothing for Performance Boost

It's to be noted, that I utilized an extra smoothing parameter in my test pipeline. In the process of testing a new review, if a word showed up in the other class, a penalty was added to the score of that class. For example, if the word 'sucks' showed up in the classification of whether the review was a positive review, a penalty would be added to the positive probability of that test review. This helped account for stop words as well. There was further a scaling coefficient that I tuned by hand with respect to each of the four models built.

Result

Confusion Matrices of each model are below along with their accuracies.

GNB_BOW: 99.9%	Actual +	Actual -
Predicted +	1498	1
Predicted -	2	1499

GNB_TFIDF: 99.8%	Actual +	Actual -
Predicted +	1496	1
Predicted -	4	1499

MNB_BOW: 99.9%	Actual +	Actual -
Predicted +	1498	0
Predicted -	2	1500

GNB_BOW: 99.9%	Actual +	Actual -
Predicted +	1498	2
Predicted -	2	1498