Naive Bayes Classification Project

This report details the methods and outcomes for CS5420 Project 2 - NB Classifier. The project was implemented in Python using the pathlib, json, re, and collections libraries. Data visualization was done using matplotlib, seaborn, sklearn.metrics, and numpy.

Problem

The goal of the project was to implement a naive bayes classifier for documents using the words contained in each document to predict the category it should be placed in. There were a total of twenty unique categories and 1000 documents in each category.

Data

The dataset for this project was the 20 Newsgroups collection from the School of Computer Science at Carnegie Mellon University. It contains twenty categories of 1000 documents. Those categories are, alt.atheism, comp.graphics, comp.os.ms-windows.misc, comp.sys.ibm.pc.hardware, comp.windows.x, misc.forsale, rec.autos, rec.motorcycles, rec.sport.baseball, rec.sport.hockey, sci.crypt, sci.electronics, sci.med, sci.space, soc.religion.christian, talk.politics.guns, talk.politics.mideast, talk.politics.misc, and talk.religion.misc.

Methods

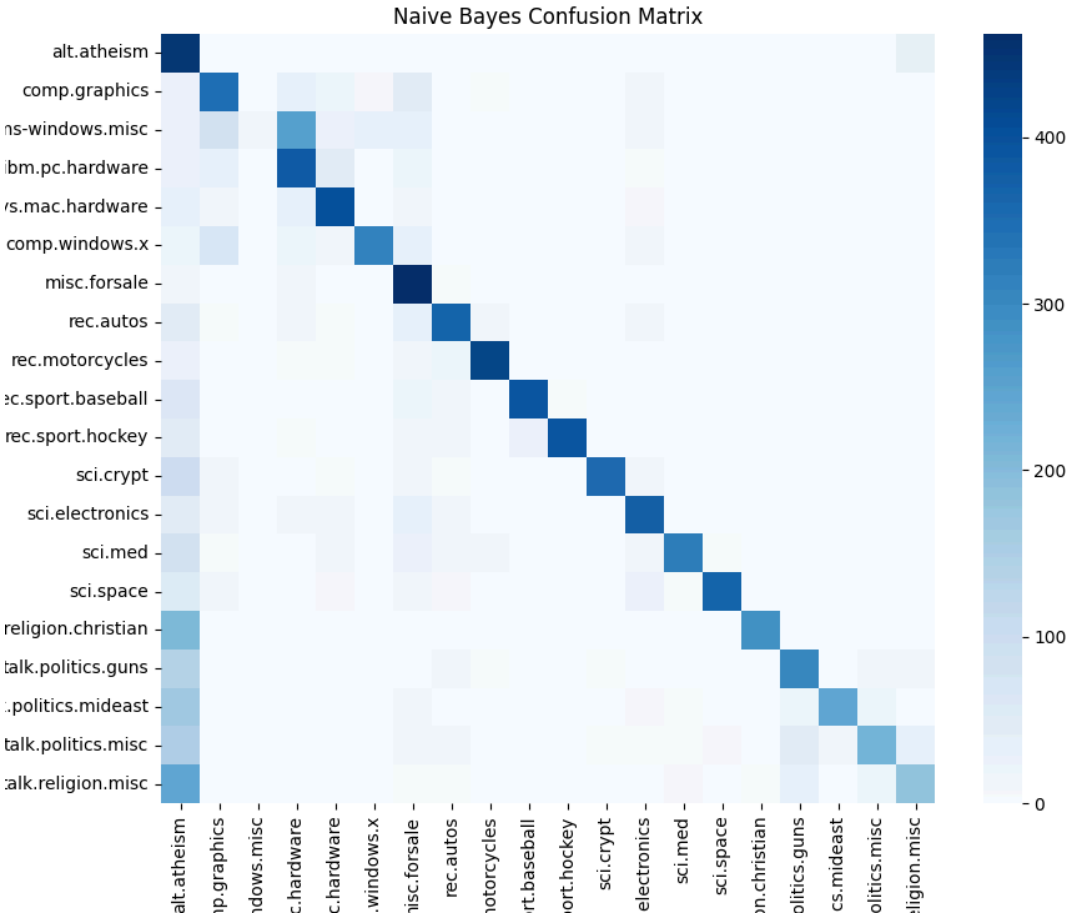
I started by splitting the dataset into 50% training and 50% testing. This was done using the pathlib and random libraries. The first step of this process was iterating through each category's directory using the iterdir() method from pathlib. Then I created a Python list of all files in the directory before shuffling them using random.shuffle(). Splitting the data was then as easy as taking the first 500 items from the shuffled list and the last 500 items. I chose to save the sets of training and testing data in separate json files so that I would not have to re-split the data on subsequent runs. I have included the training and testing indexes in my deliverable under the indexes directory, and the code used to generate them as indexData.py.

For the actual Naive Bayes implementation, I began by iterating through each file in the training set, and keeping track of the count of words within using the Counter() class. A word, for the sake of my project, was anything matching the regular expression `\b[a-zA-Z]+\b`. I then used the counts of words from each category to perform the necessary calculations of the conditional probabilities for Naive Bayes. Importantly, the prior was always equal to 0.05 as all categories were equally likely.

I did run into an issue when doing the conditional probabilities, numerical underflow. When multiplying the conditional probabilities together to get the final prediction, the numbers would become so small that python truncated them to 0, causing most documents to not be classified at all. To solve this problem, I multiplied each conditional probability by 10000 to make them large enough to not underflow. I chose 10000 by starting with 100 and adding a 0 until I stopped seeing documents classified as NONE which I used as the control category.

Results

I chose to evaluate my model by checking its overall accuracy. After classifying every document, my model came out with a 65.91% accuracy. I also made a confusion matrix using matplotlib, sckitlearn.metrics, and seaborn.



It can be observed from the confusion matrix that all categories except comp.os.ms.windows.misc were more frequently correctly classified than put in any single incorrect category. Some other patterns can also be observed, notably that alt.atheism, when incorrectly classified, was most often placed in religion.christian or talk.religion.misc, which are categories that likely have a high overlap in vocabulary.