Optimization of Classifier Performance for Sentiment Analysis in Natural Language Processing

Thomas Ray Blandford IV

Virginia Tech Pamplin College of Business

Abstract

In this study, we explore applying machine learning techniques to the task of optimizing classifier performance within the context of sentiment analysis in natural language processing. In an attempt to determine the optimal proportion of the training set relative to the total size of the corpus, we analyze key binary classification metrics to determine the accuracy of the learning model. Finally, in the context of our results, we discuss key takeaways and possible directions of further learning model optimization strategies for the natural language research community.

Keywords: Sentiment analysis, natural language processing, precision, recall, f-measure

Introduction

Deep learning, specifically in the area of natural language processing, has enormous potential for businesses to optimize their operations and provide higher quality, more individualized services to their customers. A significant amount of research in natural language processing involves developing and refining neural network models to automatically extract linguistic knowledge from online text corpora, most often from social media sources. While the number of learning variants for these problems have been increasing, the relative size of corpus divide for training, testing, and validating have been relatively unchanged. Thus, in exploring the optimal relationship of training, test, and validation set sizes within the learning model, we can hope to enhance the social utility of these systems as their computational efficiency and predictive accuracy improve.

Tools and Methods

Data Collection of Amazon Product Reviews

To begin our study, we needed to collect a data set of product reviews from Amazon. Unfortunately, my initial attempts to build a web scraper in Python proved unsuccessful. Using a product's ASIN reference number, I was able to scrape the reviews on the first page of their listing. However, Amazon's Document Object Model makes it incredibly difficult for scrapers to paginate through the entire collection of reviews, so I was unable to obtain enough data for our experiment.

Fortunately, a collection of Amazon review data is hosted by an Assistant Professor at the University of California at San Diego for use in academic projects like this one (McAuley, 2015.) For this project, we are using 10,000 reviews from the Video Game category on Amazon.

They've been cleaned and sorted by rating - positive reviews being those achieving more than 3 stars, and negative those achieving less than 3 stars.

Sentiment Analysis using Python

Following collection of our corpus, we then needed to build and train our model. Python is a simple yet powerful programming language with excellent functionality for processing linguistic data. It is the most widely used tool in the field of Artificial Intelligence, and thus, a great choice for this project.

A robust library is available in Python for text analysis known as the Natural Language Toolkit (NLTK.) It provides a naive Bayes classifier that we will use for training, testing, and validation. The naive Bayes classifier chooses a label for an input value by checking frequency of each label in the training set to calculate its probability of being positive or negative. This method for assigning probability, or score, is often referred to as a bag-of-words model.

Once the classifier has been trained, we are able to test our model and determine its performance and predictive accuracy. The most common statistical measures for analyzing the accuracy of a binary classification model are precision, recall, and f-measure. We used these as our key performance indicators in this project.

Precision

Precision tells us how exact a classifier is by describing how many inputs were labelled correctly. Precision computes the proportion of true positive classifications in the set of all positively matching label assignments (true and false positive.) This tells us out of all the inputs the model has marked as a positive match to a label, what percentage were actually labeled correctly. A high precision score means that there are fewer false positive assignments.

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

(Precision and Recall, *Wikipedia*, 2017)

Recall

Recall provides an indication of completeness to the classifier. It does this by computing the proportion of true positive label matches to all of the actual inputs that should be labeled as true (false negative and true positive.) Recall gives us an idea of how many labels were missed by focusing the classifiers attention on false negatives.

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

(Precision and Recall, *Wikipedia*, 2017)

F-Measure

F-measure is a combined measure of precision and recall. It is difficult to determine the superiority of an algorithm when using two distinct metrics. F-measure provides us a way to analyze the overall performance of these classification algorithms when we do not have a specific intent for the individual performance of precision or recall.

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

(Precision and Recall, *Wikipedia*, 2017)

Experimentation

The general consensus for splitting data within a corpus is 80% for training to 20% for testing. We've decided to test and expand this idea by training models that divide the corpus into

40:60, 60:40, 80:20, and 90:10 training and testing data sets. Each classifier was first trained, and then precision, recall, and f-measure were computed using their respective training set.

Initially, the experimentation was only going to be run on a single data set. However, following the results of the first series of tests, it seemed necessary to test with another. This new data set contained an equal number of positive and negative reviews, rather than a randomly selected bunch. This provided another useful insight in optimization of the learning model.

Results

After the experimentation on the aforementioned randomly selected data set was completed, it was apparent that negative label classification was performing very poorly in precision - scoring in the range of 2% - 20%, indicating a very high number of false positives for negative reviews. Recall for positive label classification also contained a high degree of variance, from 41% - 61%, indicating a relatively large number of false negatives. Overall, it seems the most optimal split for this data set was 60:40 based on f-measure scores. See Figure 1.

To determine if the poor negative performance was related to a lack of negative reviews in the data set, it seemed necessary to add another experiment that accounted for training on an equal number of negative and positive reviews. The experiments were reconducted, this time with a data set of 5000 positive reviews and 5000 negative reviews.

This did seem to have a noticeable effect on negative precision, boosting the scores to a high of 73% on an 80:20 split. There seemed to be more extreme variances in the tests, however, as negative precision was as low as 8% on a 90:10 split. Positive precision was slightly lowered in this series of tests, but f-measure for both labels were significantly improved. See Figure 2.

Overall for the second data set, it seems that the consensus rule of 90:10 was the most optimal algorithm with the highest positive and negative f-measures of 79% and 85% respectively. This improved on the positive and negative f-measure scores from the randomly selected data by 12% and 56% respectively. This data suggests that for a data set of 10,000 Amazon reviews, using a collection of equally positive and negative reviews for the corpus with a 90:10 split of training and testing data would be the most accurate.

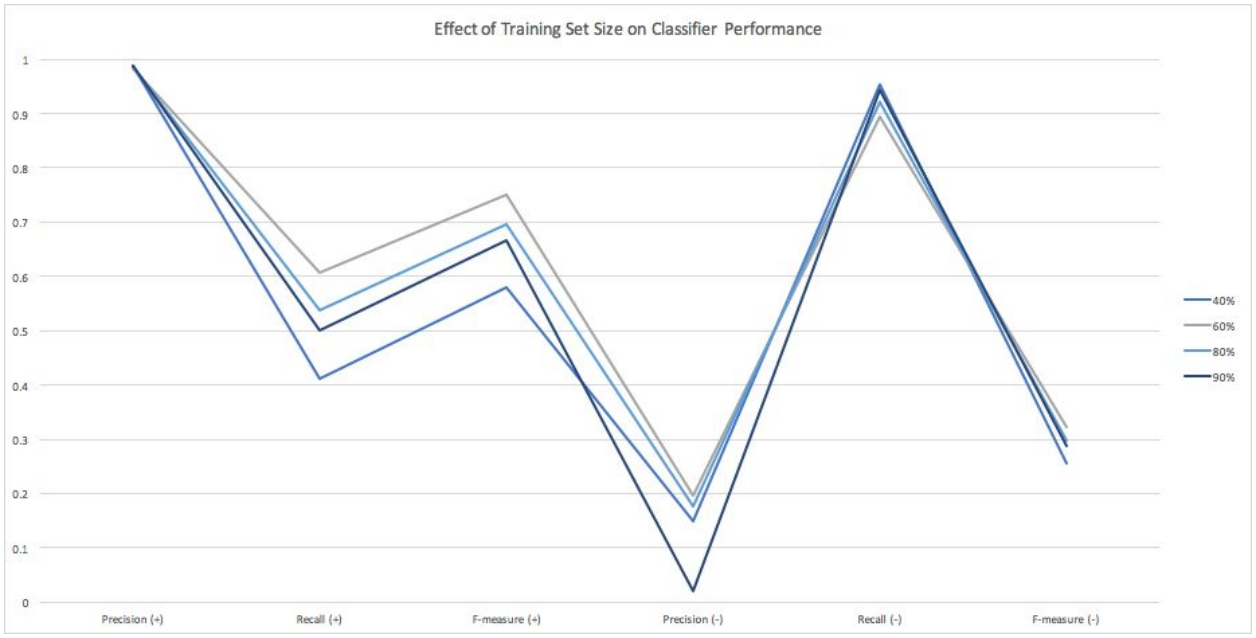

Figure 1 - First series of test when training on randomly selected product reviews. This figure demonstrates the relationship of precision, recall, and f-measure to training set size on both positive and negative labels.

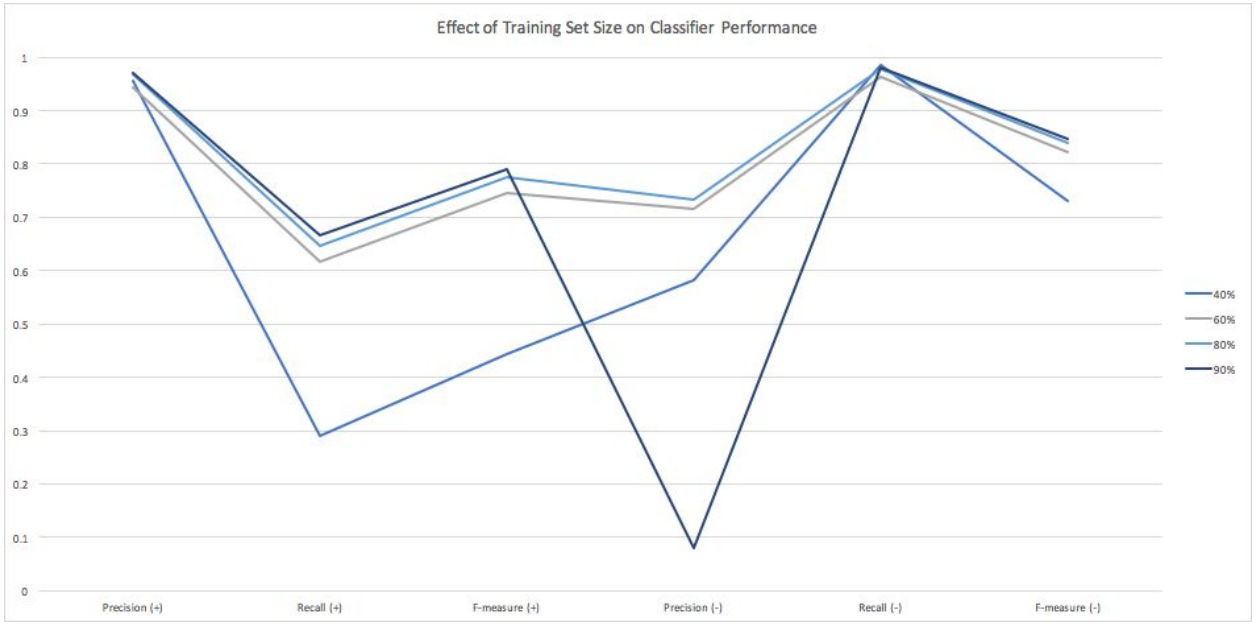

Figure 2 - Second series of tests when trained on equal number of positive and negative product reviews. This figure demonstrates the relationship of precision, recall, and f-measure to training set size on both positive and negative labels.

Discussion

This project has taught me a lot about how artificial intelligence and computational linguistics converge, and have deepened my knowledge in the fields of statistics and computer science. I began with a rudimentary background in Python, deep learning, and natural language processing. I would now classify myself as proficient in Python, and have a fundamental understanding of training neural network models, and analyzing the effectiveness of a naive Bayes classifier using scores of precision, recall, and f-measure.

Additionally, I have found there are many variables that can be tweaked to improve the efficiency of a learning algorithm - whether the goal is to achieve optimal overall performance via f-measure, or by targeting an optimal precision or recall score. Potential areas for further

study still exist in the size of the data set. What is the relationship of these algorithmic measures to the corpus size? Some of the observed variance can potentially be limited via a larger or smaller collection of reviews. The models should also be optimized for double negatives, like "not bad," which are perceived by humans as positive but by the algorithm as negative. Lastly, we can further train our model for studies in natural language generation, named entity recognition, text classification, or information extraction.

Natural Language Processing has huge implications on the way that humans and computers interact by providing a bridge between human communication and digital data. This technology will revolutionize many industries with machine translation, summarization, sentiment analysis, and many other applications. The discoveries of this project aim to support research in this complex field by providing some insight on building a robust model optimized for predictive accuracy.

References

Bird, S. (2016). *Natural language processing with python*. OReilly Media.

Goldberg, Y. (2015). A Primer on Neural Network Models for Natural Language Processing. Retrieved December 20, 2017, from http://u.cs.biu.ac.il/~yogo/nnlp.pdf

Manning, C., & Socher, R. (n.d.). CS224n: Natural Language Processing with Deep Learning. Retrieved December 20, 2017, from http://web.stanford.edu/class/cs224n/

McAuley, J., Targett, C., Shi, J., & Van den Hengel, A. (2015). *Image-based recommendations on styles and substitutes.* Retrieved December 18, 2017, from http://cseweb.ucsd.edu/~jmcauley/pdfs/sigir15.pdf

Precision and Recall. (n.d.). In *Wikipedia.* Retrieved December 18, 2017, from https://en.wikipedia.org/wiki/Precision_and_recall

Vanhoucke, V., & Chakraborty, A. (n.d.). Deep Learning By Google. Retrieved December 20, 2017, from https://www.udacity.com/course/deep-learning--ud730