Machine Learning Engineer Nanodegree
Capstone Proposal

Ritu Agarwal
August 31, 2018

A comparative study of Sentiment Analysis using various ML algorithms

Domain Background
With the democratization of the Internet every individual now has an opportunity to express his or her opinion on any topic, event, company, indeed anything at all. They have an opportunity to express themselves through pictures, words, tweets, posts and numerous other ways.
The challenge then lies in assessing the emotions, feelings, passion, sentiments being conveyed by these individuals or the population as a whole. Given the large volumes of data, the only possible way to really assess these sentiments is computationally. Sentiment analysis is this process of examining the data and drawing conclusions of what the emotions are of the population on any given topic.

This project is going to analyze the sentiments of movie reviews from a Rotten Tomatoes dataset using various ML and NLP algorithms.

Problem Statement
The goal of this project is to implement and train various ML and NLP algorithms to predict the sentiment on a Rotten Tomatoes movie reviews dataset provided by Kaggle [1] and compare them against a preset metrics to figure out which provides the best accuracy. For each algorithm the data would have to be pre-processed, split into training and validation sets, the classifier would then be trained on the training set and the model would then be tested on the validation set.

This effort aspires to defining a model that is most optimal for datasets and problem statements that this project tackles. The hope is that future users will have to contend with fewer choices and possibly improve upon the work done here.

Datasets and Inputs

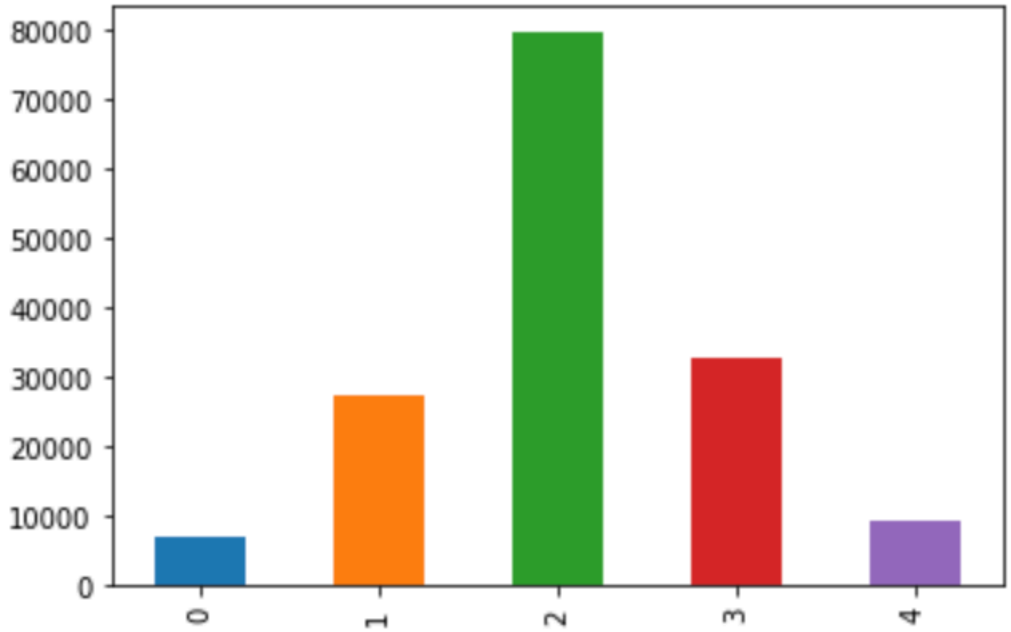
The dataset for this project is a Rotten Tomatoes movie review dataset provided here [1]. It is a corpus of movie reviews used for sentiment analysis, originally collected by Pang and Lee [4]. In their work on sentiment treebanks, Socher et al. [5] used Amazon's Mechanical Turk to create fine-grained labels for all parsed phrases in the corpus. A working model can be seen at [6]. The phrases are labelled on a scale of five values: negative, somewhat negative, neutral, somewhat positive and positive. There are obstacles like sentence negation, sarcasm, terseness and language ambiguity that make this task very challenging.

The data is in a tab separated file 'train.tsv'. The phrases all come from the Rotten Tomatoes dataset. Each Sentence has been parsed into many phrases by the Stanford parser. Each phrase has a PhraseId and each sentence has a SentenceId. Phrases that are repeated (such as short/common words) are only included once in the data.
Each phrase has a sentiment label associated with it as follows:
0 - negative
1 - somewhat negative
2 - neutral
3 - somewhat positive
4 – positive

Total number of Phrases – 156060
Phrases Count by label:

Phrase	Count
Negative	7072
Almost Negative	27273
Neutral	79582
Almost Positive	32927
Positive	9206

A plot of the Phrases is shown below –



For the project the data will be split into training and validation set by an 80-20 ratio.

Solution Statement

The first step will be to create a Benchmark model using CountVectorizer [3] to create a "Bag of Words" model [2] and train it using a Naïve Bayes classifier to make predictions on the validation set [8].

Next steps will be to build and test the same data on a couple of more Supervised learning models like Support Vector Machine and some ensemble methods like RandomForest classifier to see if the accuracy is any better.

Then build more models like a Convolutional Neural Network architecture, a Recurrent Neural Network architecture using Long Short-Term Memory layer, a Word2Vec in python (inspired by a Kaggle tutorial at [7]) and see if the performance can be improved upon.

Benchmark model

The Benchmark model will be generated using CountVectorizer from sklearn package to create a Bag of Words model which will then be trained using a Naïve Bayes classifier on the given training data. The predictions will then be made on the validation data and results tested using the accuracy score and the F1 score.

Evaluation Metrics

A couple of Metrics will be used for Evaluation -

The goal is to reach a high *accuracy score* [8, 9, 10] in predicting the correct sentiment of a movie review that was written on Rotten Tomatoes. The simplest definition of an accuracy score is the proportion of correctly classified objects. Since the higher the number of correctly classified objects, the better the model, accuracy score is the metrics that will be used to find the most suitable algorithm for the project.

Since this is an unbalanced multi-class classification problem, it is better to have more than one metric for evaluation and the project will also use *F1 score* to compare the various models. For neural networks, Keras will be used and it no longer support F1 score in its metrics. An implementation inspired by [11] will be coded.

Project Design

The following steps will be followed in the project –

1. Data Analysis and Cleansing: In the first step the data and its various features will be thoroughly examined. The sentiment spread will be studied and analyzed. The different phrases will be cleaned by tokenizing, removing the stop words, punctuation and uppercase letters and then Vectorizing the results. Experimenting will be done with CountVectorizer, TfidfVectorizer, and features needed for Word2Vec vectors. The data will be split into train and test set.
2. Train the Benchmark Naïve Bayes classifier and produce the benchmark results.
3. Train an SVM classifier and a RandomForest classifier and test the performance.
4. Set up a Convolutional Neural Network and train the model. Try fine tuning the CNN by adding, dropping layers to get the best possible accuracy results.
5. Setup an RNN using an LSTM layer and train the model. Try fine tuning the model to get the best possible accuracy.
6. Setup a Word2Vec model and calculate the accuracy.
7. Compare and contrast the accuracy and F1 score of all the models to draw conclusions on which model is the best suited for Sentiment Analysis.

Resources

[1] https://www.kaggle.com/c/movie-review-sentiment-analysis-kernels-only/data
[2] https://en.wikipedia.org/wiki/Bag-of-words_model
[3] http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html
[4] Pang and L. Lee. 2005. *Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales*. In ACL, pages 115–124.
[5] Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank, Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Chris Manning, Andrew Ng and Chris Potts. Conference on Empirical Methods in Natural Language Processing (EMNLP 2013).
[6] http://nlp.stanford.edu/sentiment/
[7] https://www.kaggle.com/c/word2vec-nlp-tutorial
[8] *Investigating the Working of Text Classifiers,* Devendra Singh Sachan,4, Manzil Zaheer, Ruslan Salakhutdinov, School of Computer Science, Carnegie Mellon University 4Petuum, Inc Pittsburgh, PA, USA https://arxiv.org/pdf/1801.06261.pdf
[9] R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning. Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions. In EMNLP, 2011. http://www.aclweb.org/anthology/D11-1014
[10] *Multiclass Sentiment Prediction using Yelp Business Reviews,* April Yu, and Daryl Chang, Dept. of Computer Science, Stanford University. http://cs224d.stanford.edu/reports/YuApril.pdf
[11] https://medium.com/@thongonary/how-to-compute-f1-score-for-each-epoch-in-keras-a1acd17715a2