SENTIMENT CLASSFICATION ON PRODCUT REVIEWS

Abhijit Ravindra Balihalli
Siddharth Waghela
Nikhil Keshav Bhoir

October 27, 2019

Table of Contents

1 Introduction

The aim of this challenge is to develop a sentiment classifier that can assign a large set of product reviews to the five levels of polarity of opinion as accurately as possible, given a small amount of labeled reviews and a large amount of unlabelled reviews. It is a multi-class classification task, where each product review is labeled with one of the five sentiment labels, which are strong negative, weak negative, neutral, weak positive, and strong positive

2 Pre-processing and feature generation

"Data pre-processing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviours or trends, and is likely to contain many errors. Data pre-processing is a proven method of resolving such issues." (Sharma, 2019)
"Feature engineering is the process of using domain knowledge of the data to create features that make machine learning algorithms work." (En.wikipedia.org, n.d.)
For this project, our team first performed feature generation, where the raw, unstructured data is taken and features are generated after performing data pre-processing activities. During feature generation, stop words and words containing three or lesser characters were removed and also bi-gram features was generated.

2.1 Pre-processing

There are multiple ways of performing data pre-processing while performing Sentiment Analysis. The data pre-processing steps performed before feature selection for the Yelp product reviews are listed below.

2.1.1 Unicode

Before performing other cleaning activities on the dataframe, encoding and decoding the strings was performed as the strings are stored as Unicode and every character has a code associated with it. But some special characters whose Unicode cannot be coded efficiently, so we performed encoding on the 'text' field and ignored those characters with unencodable unicode and converted the text back to 'utf-8' format by decoding.

2.1.2 Lowercasing

Lower casing the data is one of the most simple and effective way to perform data pre-processing. It helps with maintain consistency throughout the dataframe, especially given that the data contains reviews by different people and every person has a different way of typing. So, converting all the text to either lower case or uppercase is important to maintain consistency throughout the dataframe. We decided to go with lowercasing as this is the most preferred way.

2.1.3 Removing Special and Newline Characters

Since each datapoints contains reviews from different people and contain multiple sentences within each review, removing newline characters is important for further cleaning activities. Also, we retained only characters and numbers and removed all the special characters. Removing this level of

noise is important before performing stemming activity, else some characters from the words might be deleted when stemming is performed.

2.1.4 Stemming

"Stemming is the process of reducing a word to its word stem that affixes to suffixes and prefixes or to the roots of words known as a lemma. Stemming is important in natural language understanding (NLU) and natural language processing (NLP)" (Rouse, n.d.).

Given the reviews are coming from different people, and also searches are made for different set of words, stemming is definitely necessary to be performed as part of cleaning activity.
To perform stemming, we split the sentences into individual words and used the function 'stemmer.stem()' on each word. If the word matches with the root word, then the word is converted to the root word and stored in a list. After running through the entire comment word-by-word, it is converted back into a sentence and stored back in the 'text' column.

2.2 Feature Engineering

Feature Engineering is very important when dealing with text data, as the text contains a lot of information, some useful and some not as much.

2.2.1 Feature Set 1

The team performed feature engineering by performing 'Bi-gram' feature, removing stop words, removing words with three or less characters. All the generated features were used to build the model for predicting labels on the test data.

- **Bi-gram Feature**
  "… n-gram is a contiguous sequence of *n* items from a given sample of text or speech" (En.wikipedia.org, n.d.).
  For our project, we selected Bi-gram feature where it takes 2 items which occur in a sequence. This was performed by using the 'ngram_range' parameter within the 'CountVectorizer' function which is found in the 'sklearn' library.
- **Removing Stopwords**
  "Stop words are a set of commonly used words in a language. Examples of stop words in English are "a", "the", "is", "are" and etc." (Ganesan, n.d.). Stop words provide very low information and is not considered very important for NLP. Instead the importance has to be given for important words.
  Stop words are removed before performing Vectorization by selecting the most commonly used words in English language. Stop words data is found in the 'nltk' library.
- **Removing Words with Fewer Characters**
  Words which have fewer characters like three or lesser characters are not very helpful in sentiment analysis. If such words are not removed while removing stop words with the same length, then it is a good practice to remove them at this stage manually as most of them form part of stop words.

2.2.2 Feature Set 2

For the second feature set, feature engineering was performed in R. The team performed feature engineering by performing removing stop words. All the generated features were used to build the model for predicting labels on the test data.

- **Removing Stopwords**
  "Stop words are a set of commonly used words in a language. Examples of stop words in English are "a", "the", "is", "are" and etc." (Ganesan, n.d.). Stop words provide very low information and is not considered very important for NLP. Instead the importance has to be given for important words.
  Stop words are removed before performing Vectorization by selecting the most commonly used words in English language. Stop words data is found in the 'tm' library.

3 Models

For the model development or classification algorithm there are many statistical models available to perform sentiment analysis. The team decided upon three models to train on the labeled and unlabeled datasets and finally do the prediction on the test dataset.
The models chosen are – Logistic Regression, Multinomial Naïve Bayes and Deep Learning using Neural Networks.

3.1 Logistic Regression

Logistic Regression is highly used supervised machine learning algorithm. It is mainly used for Binary Classification. But in our project, we are using Multinomial Logistic Regression. "…**multinomial** logistic regression is a classification method that generalizes logistic regression to multiclass problems, i.e. with more than two possible discrete outcomes" (En.wikipedia.org, n.d.).

Multinomial Logistic regression is selected instead of binary logistic regression because the target variable we are training contains multilevel (5 labels). Logistic regression is not only easy but it is a good practice to have this model as the base model and compare the other more complex models against the logistic model.

3.2 Multinomial Naïve Bayes

Naïve Bayes is a group of generalized algorithms which are based on Bayes theorem. The model assumes that each feature it uses are independent conditionally to one another.

$$p(f_1, \ldots, f_n|c) = \prod_{i=1}^{n} p(f_i|c)$$

Here, f1…fn are features under some class c. In our classification case, bag of words that we generate are the features and the labels are the classes.
It calculates probability, i.e. given a class of features, what is the probability that it belongs to a class.
Naïve Bayes models have performed well, where strong dependence between the features is false.
The term, Multinomial Naïve Bayes simply means the probability distributions are multinomial in nature. In natural language processing, multinomial distributions and categorical distributions are similar. Hence, we opted to use Multinomial Naïve Bayes for our text classifier.

3.3 Deep Learning using Neural Networks

The deeplearning model was developed in R. For developing this model "h2o" library was used. H2O deeplearning model is a multi-layer feedforward artificial neural network. And stochastic gradient

descent is used to train the neural network using back-propagation. This model can contain multiple hidden layers and multiple neurons. Also, it has different activation functions like tanh, rectifier, and maxout. Apart from this, different parameters like learning rate, L1 or L2 regularization, rate annealing, momentum training can be set in order to get better prediction accuracy. Each compute node trains a copy of the global model parameters on its local data with multi-threading (asynchronously) and contributes periodically to the global model via model averaging across the network.

3.4 Discussion of model difference(s)

Model	Advantages	Disadvantages
Logistic regression	Multinomial Logistic regression is easy and efficient to implement and train	
Naïve Bayes	• During feature generation for text classification, words are not related to each other, thereby supporting the assumption of Naïve Bayes. • Training is quick and less computational power is required than Deep learning algorithms	Fails to perform when there is a dependency between the features.
Deep Learning	Neural network can generate non-linear decision boundaries. This means that if the data is not linearly separable then other algorithms will not be classifying the data with better accuracy. On the other hand, neural networks can learn the non-linear decision boundaries through multiple neurons, feedforward and back propagation	As it involves a lot of computation, it generally works slow and takes a lot of time for large datasets.

4 Experiment setups

4.1 Feature Set 1

1. Pre-processing and cleaning the labeled, unlabeled and test data. – It means that we are preparing our data and cleaning it for further features generation.
   - It includes removing uncodeable characters, retaining rest of the data, new line characters, unwanted space, special characters, stemming of the data, basically trims the words, hence extracting the original words.
2. Creating features using Count Vectorizer, means with the clean text we will be generating features. First tokenizing the text and represent each word as a feature.
3. Applying Multinomial Logistic regression to the labeled data by using sklearn's library LogisticRegression, it predicts the label based on the probabilities.
4. Retraining the model on labeled and certain % of predicted unlabeled data.
5. Lastly, implementing this model on the given test data.

The above same steps are followed for Multinomial Naïve Bayes mode.

4.2 Feature Set 2

The deeplearning model was developed in R. For developing this model "h2o" library was used. For using the algorithms in "h2o" library, data frames were converted to H2OFrames. Then following steps were performed.

1. **Auto-Encoder:** Autoencoder is an unsupervised artificial neural network that learns how to efficiently compress and encode data then learns how to reconstruct the data back from the reduced encoded representation to a representation that is as close to the original input as possible. Using the autoencoder we got the additional features for the data. To train the autoencoder, labelled data was used as training_frame. Number of hidden layers were 1 and number of units in the hidden layer were 100. Epoch were set to 50 and activation function was set as tanh.
2. **Generating Additional features:** Then using the auto-encoder, additional features were generated for labelled, unlabelled and test dataframes. And these additional features were added to the respective dataframes.
3. **Classifier based on labelled data:** Using h2o deeplearning, classifier was constructed for labelled data. For this labelled dataframe was used as training_frame. One hidden layer was used with 10 units, 50 epochs, activation function as tanh and L2 regularization as 0.1.
4. **Predict labels for unlabelled data:** predicted labels for unlabelled data were generated using the classifier which was build in last step.
5. **Combine labelled and unlabelled data:** After generating the predicted labels for unlabelled data, both labelled data and unlabelled data were combined using rbind. This data was then used to train the model again
6. **Classifier based on combined data:** Using h2o deeplearning, classifier was constructed for combined data. For this combined dataframe was used as training_frame. One hidden layer was used with 10 units, 50 epochs, activation function as tanh and L2 regularization as 0.1.
7. **Predict labels for test data:** Using the classifier build in last step, the labels for test data were predicted.

5 Experimental results

Model	Accuracy (%)
Logistic Regression	58.26
Naïve Bayes	43.06
Deep Learning	52.75

6. Conclusion

Feature Generation is important for Sentiment Analysis on large set of text data. Here in this project we have generated two feature sets. The first feature set has count vector with unigrams, bigrams and removing stop words, few letter words and the other feature set having unigrams and stop words being removed.

We observed that Logistic Regrssion model had the best accuracy among the three-model chosen. We cannot say it is the best model, because the features used for Deep Learning algorithm are different from logistic regression model. But in this scenario, it produced the best accuracy.

Given better resources with higher processing power, all the data points could be used and better feature selection can be performed for a better accuracy. Semi supervised learning resembles real life scenario data with abundant unlabelled data and very few or in fact non-existent labelled data.

References

Ganesan, K. (n.d.). *All you need to know about text preprocessing for NLP and Machine Learning - KDnuggets*. [online] KDnuggets. Available at: https://www.kdnuggets.com/2019/04/text-preprocessing-nlp-machine-learning.html

En.wikipedia.org. (n.d.). *Feature engineering*. [online] Available at: https://en.wikipedia.org/wiki/Feature_engineering

Rouse, M. (n.d.). *What is stemming? - Definition from WhatIs.com*. [online] SearchEnterpriseAI. Available at: https://searchenterpriseai.techtarget.com/definition/stemming

Ganesan, K. (n.d.). *All you need to know about text preprocessing for NLP and Machine Learning - KDnuggets*. [online] KDnuggets. Available at: https://www.kdnuggets.com/2019/04/text-preprocessing-nlp-machine-learning.html

En.wikipedia.org. (n.d.). *N-gram*. [online] Available at: https://en.wikipedia.org/wiki/N-gram

En.wikipedia.org. (n.d.). *Multinomial logistic regression*. [online] Available at: https://en.wikipedia.org/wiki/Multinomial_logistic_regression

http://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/deep-learning.html

https://towardsdatascience.com/auto-encoder-what-is-it-and-what-is-it-used-for-part-1-3e5c6f017726