Engineering Applications of Artificial Intelligence

To Study the Effect of Different PreProcessing Techniques on Classification Models in NLP

--Manuscript Draft--

Manuscript Number:	EAAI-21-66
Article Type:	Research paper
Keywords:	natural language processing; Classification models; Pre-processing; data.
Corresponding Author:	Dhnesh Dhingra, Btech JMIT: Seth Jai Parkash Mukand Lal Institute of Engineering and Technology INDIA
First Author:	Dhnesh Dhingra, Btech
Order of Authors:	Dhnesh Dhingra, Btech
Abstract:	A lot of Machine Learning Models Studies focus on Optimizing Accuracy, enhancing model performance, optimizing parameters. AI depends heavily on data. And on top of that, if your data is not processed precisely, the model performance would not be up to the standards. This study discusses the combinations of preprocessing types in a text-processing neural network model. In this study various classification models like Naive Bayes, KNNs are applied on Amazon Product Review Dataset. We shuffled various Pre-processing techniques on the data to evaluate and compare the accuracy of the final model. Finally, our experiment provides insights into the best preprocessing practices for training NLP models and word-embeddings.

AUTHOR DECLARATION TEMPLATE

We wish to draw the attention of the Editor to the following facts which may be considered as potential conflicts of interest and to significant financial contributions to this work. [OR]
We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed. We further confirm that the order of authors listed in the manuscript has been approved by all of us.

We confirm that we have given due consideration to the protection of intellectual property associated with this work and that there are no impediments to publication, including the timing of publication, with respect to intellectual property. In so doing we confirm that we have followed the regulations of our institutions concerning intellectual property.

We understand that the Corresponding Author is the sole contact for the Editorial process (including Editorial Manager and direct communications with the office). He/she is responsible for communicating with the other authors about progress, submissions of revisions and final approval of proofs. We confirm that we have provided a current, correct email address which is accessible by the Corresponding Author.

Signed by all authors as follows:

Dhnesh 23 Jan 2021

To Study the Effect of Different PreProcessing Techniques on Classification Models in NLP

Abstract:

A lot of Machine Learning Models Studies focus on Optimizing Accuracy, enhancing model performance, optimizing parameters. AI depends heavily on data. And on top of that, if your data is not processed precisely, the model performance would not be up to the standards. This study discusses the combinations of preprocessing types in a text-processing neural network model. In this study various classification models like Naive Bayes, KNNs are applied on Amazon Product Review Dataset. We shuffled various Pre-processing techniques on the data to evaluate and compare the accuracy of the final model. Finally, our experiment provides insights into the best preprocessing practices for training NLP models and word-embeddings.

Keywords: Natural Language Processing, Classification models, Pre-processing, data.

Introduction

We as humans have no clue about when the language was invented, how it was created. To be more precise what language is. There are many questions like was it developed at a certain place and spread with the evolution of humans all across the earth. We simply don't have the foggiest idea. Human communication isn’t just words and their explicit meanings. Instead, it’s nuanced and complex. Speech is the most significant and effective mode of communication. Humans train the animals like dogs, cats with voice orders, forcing on to them our communication mechanism with reward signals when they respond or act. But we are trying to enable computers to understand everyday human speech or ordinary written language. Algorithms can't value emotions, but NLP(Natural Language Processing) disagree with this. Google created an AI language model called BERT (Bidirectional Encoder Representations from Transformers) which also demonstrated higher accuracy than humans in the challenging area of question answering. In the late 1940s, Alan Turing wrote a paper explaining a test for a “thinking” machine. He analyzed that if a machine could be part of a conversation with the help of a teleprinter, and it imitated any human so exactly there were no detectable differences, then that machine could be considered as capable of thinking. Natural language processing (NLP) is a field of artificial intelligence in which smart engines analyze, understand, and extract meaning from human language in a smart and valuable way. For executing, this field of AI like any other field depends heavily on data. The characteristics of the data used in machine learning are an important factor in determining the efficiency of learning.

And data comes with various anomalies and the following issues:

- Any inconsistencies and/or errors existing in the data.
- Any duplicates or outliers in the data.
- Data integration problems
- Any normalization or other transformation of the data.
- Contradicting data
- Any derived data needed for the analysis.

If data is not preprocessed, the algorithm may behave unexpectedly due to inconsistent data, and performance may be affected.

So how do we process the data? Say we have lots of textual data corpus to study or analyze. And this textual data is processed in different refinement techniques which we will discuss. Then this refined data is converted into a form that will be easier to acquire knowledge from the given text. This method/process is titled "text-preprocessing.". In Section 2 we will discuss the techniques most commonly used for preprocessing. In Section 3 we will discuss and analyze the experiments done and discuss the limitations of over-using these techniques.

Section 2

Let's see the various different procedures or techniques that are followed while preprocessing the data also used for dimensionality reduction.

1. Lower casing

Suppose you created an NLP model without working on lowering every word in a sentence. Now, what if a word which is at the beginning of a sentence with a capital letter is the same word that appears later in the sentence but without any capital letter. Your model will treat them two different words while parsing. Which is not a good way to train any model. For Example, It will allow instances of "Hello" at the beginning of a sentence to not match with a query of "hello". Thus, Converting all your data to lowercase helps in the process of preprocessing and in later stages in the NLP application, when you are doing parsing.

2. Tokenization

Tokenization is basically splitting a sentence, expression, phrase, paragraph, or a complete book text document into tiny units, for example, singular/individual words or terms. Each one of these tiny units is called tokens. To be more precise tokenization is the task of chopping it up into pieces, called tokens
The most widely recognized method of framing tokens depends on space. Consider a sentence "He is a good friend of mine."

Accepting space as a delimiter, the tokenization of the sentence brings about 7 tokens – He-is-a-good-friend-of-mine. As each token is loosely referred to as a word, it turns into an illustration of Word tokenization. A token is an example of a succession of characters in some specific documents that are grouped together as a significant semantic unit for data handling or preprocessing. This explains the importance for tokenization

3.Stop words removal

There are words for example in English like 'the', 'is', 'there', 'a', which are not significant and termed as stop words. Generally these words are not processed. If stop words are not handled to be skipped/removed, , then this will take up additional space in the database or memory. This way, the efficiency of the code reduces by a great extent. The list of stop words is long. The intuition behind using stop words is that, by removing low information words from text, we can focus on the important words instead. Most of the time, the stop word list for the given language is a well hand-curated list of words that occur most commonly across corpus.

4.Stemming

As Wikipedia says stemming is the process of minimizing inflected (derived) words to their base or root form, word stem precisely. For example "wait", "waited", "waiting", "waits" would be stemmed to "wait". When a form of a word is recognized it makes sense to get its base form to make it possible to match with a similar word having in their root form or not. Stemming is a crucial part of the pipelining process in Natural language processing
Similarly, if we have the words — liked, likely, and liking — we can apply stemming algorithms to get the root word — like. Search engines heavily used stemming for query fetching and finding domain vocabularies in domain analysis. There are many famous libraries developed universally for Stemming like Porter Stemmer, Snowball Stemmer, Lancaster Stemmer.

5.Lemmatization

Having described stemming, we can assume the definition of Lemmatization is similar to stemming but with the inclusion of context. Lemmatization considers the context and mold the word to its significant base form, whereas stemming simply eliminates the last couple of characters, frequently prompting incorrect meanings and spelling blunders. Let's take words like "history", "historical" if applied lemmatization it will return them as "history". But if we run stemming on the same words it will return the base form like "histori". Likewise for the words "finally, final, finalized" when stemmed returned "fine", but when lemmatized will give the output as "final". So you see lemmatization makes sense as compared to stemming. As the word produced by stemming "histori" and "fine" does not make sense and does not consider the contextual form. Lemmatization is a more powerful operation, and it takes into consideration morphological analysis of the words. It returns the lemma which is the base form of all its inflectional forms.

Section 3

In the last section we see how the various textual-data processing methods are used and their importance.
But overusing them or using all of the mentioned procedures could have a bad impact on the accuracy of our model.
These prepossessing steps may affect the overall performance of the detection algorithm.

This study is applied to various combinations of preprocessing techniques, and these techniques were studied separately. The dataset is of Amazon Product Reviews from "https://s3.amazonaws.com/fast-ai-nlp/amazon_review_polarity_csv.tgz". The Polarity analysis was categorized as taking review score 1 and 2 as negative, and 4 and 5 as positive. A Sample of score 3 is ignored. Dataset is split into 1,200,000 training and 80,000 testing samples. With respect to labels, there are two classes. Class 1 is a negative class 2 is positive.

```
     label          review_text
0     2     Excellent product, super quality. It capable to play audible audiobooks and it
reads kindle e books
1     1     was expecting better sound quality as size increased.. At least more Bass can be
added.
2     2     Amazing!     This soundtrack is my favorite music of all time...
3     2     Excellent Soundtrack  I truly like this soundtrack and I enjoy the video...
4     1     Paid more money for nothing. Very disappointed with the product
```

To test this, we experimented with different classification algorithms to evaluate the impact of preprocessing.

Naive Bayes Classifier: A Naïve Bayes classifier is a simple algorithm for classifying data based on Bayes' theorem. Generally, it is considered to be a supervised algorithm that uses the below equation (Bayes' theorem) to calculate the probability of a specific event occurring given a specific set of features values. P(predictor | target): is the probability of this target value giving

$$\text{Posterior} = \frac{\text{Likelihood* Prior}}{\text{Evidence}}$$

$$P(target| predictor) = \frac{P(predictor \mid target) \,.\, P(target)}{P(predictor)}$$

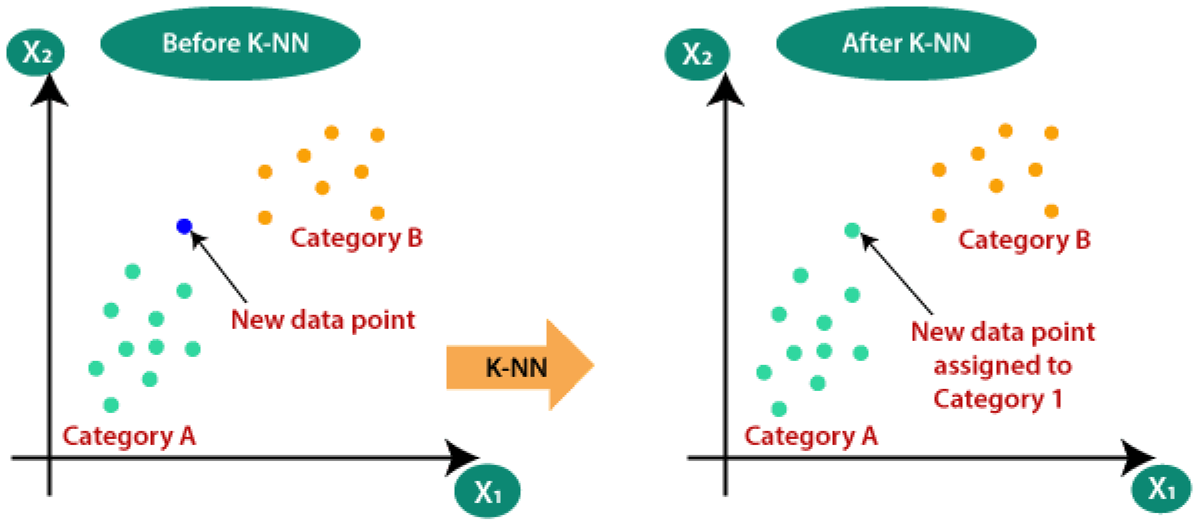
this predictor value.

P(target): is the probability of target value.
P(predictor): is the probability of this predictor value.
P(target| predictor): is the predicted probability of target giving predictor.

K-Nearest Neighbor(KNN): K-Nearest Neighbour is also a Supervised Learning technique. K-NN algorithm assumes the similarity between the new case/data and available cases and puts the new case into the category that is most similar to the available categories. It is a non-parametric algorithm, which means it does not make any assumption on underlying data. When implementing KNN, the first step is to transform data points into feature vectors, or their mathematical value. The algorithm then works by finding the distance between the mathematical values of these points. The most common way to find this distance is the Euclidean distance, as shown below.

$$d(\mathbf{p},\mathbf{q}) = d(\mathbf{q},\mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$
$$= \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}.$$

KNN runs this formula to compute the distance between each data point and the test data. It then finds the probability of these points being similar to the test data and classifies it based on which points share the highest probabilities.



The results of applying stemming and punctuation marks removal preprocessing

	With Stemming and Removing Stop Words	Without Stemming	Removing Stop Words
	Accuracy.	Accuracy	Accuracy
NB	78.5%	79.7%	76.3%
KNN	74.8%	72.6%	70.4%

The effect of applying various preprocessing methods was analyzed according to various criteria. We can notice without stemming Naive Bayes performs the best with 79.7% whereas KNN performs 72.6%. On the other hand, If we remove the stop words the accuracy drops down to 76% which is considered a major drop and the same happens in K-Nearest Neighbor Classifiers. This suggests that the preprocessing decisions are not so important when the training data is large enough, but they are indeed relevant in benchmarks where the training data is limited.

Lowering Conversion to lowercase
Advantages: search precision can be improved
Disadvantages: proper nouns composed of capital letters can be incorrectly classified as nouns

Stemming Advantages: time efficiency can be improved by reducing the size of the text
Disadvantages: dilution of meaning can affect the accuracy

Lemmatization Conversion to headwords
Advantages: part-of-speech information is converted into a preserved form, and search accuracy can be improved
Disadvantages: conversion time is long

Punctuation
Splitting Word splitting Advantages: meaning can be preserved
Disadvantages: different rules should be applied depending on the purpose, and the rules are complicated

In light of the investigation of the outcomes, we verified that a mix of two preprocessing strategies demonstrated great execution. The variance of the normal estimation of the outcomes by the preprocessing type went from at least 0.02 to a limit of 0.44. If only two preprocessing techniques can be used, lemmatization and punctuation splitting are good candidates. We observed that the current definition of the coherence metric is ineffective for the task of stopword removal. We know that Stopwords are non-informative or noise tokens that can be removed during any preprocessing step. But we noticed that common words are mostly found in high frequency on all topics or documents. The removal of which drops coherence significantly. Lemmatization and lowering are techniques that can improve accuracy by normalizing different words.
This implies that splitting a word into two words based on punctuation, such as during normalization or punctuation splitting, can extend the length of a sentence and have a positive effect on accuracy.
Our evaluations highlight the importance of being careful in the choice of how to preprocess our data and to be consistent when comparing different systems.

Conclusions

In this paper, we discussed the importance of NLP and then the various types of Preprocessing techniques that are commonly used in NLP. The most common ones are Lower casing, Tokenization, Stop words removal, Stemming, Lemmatization. Then we investigated the effects of applying many different standard NLP preprocessing steps on the performance of many different supervised learning reviews classifications. While we know the benefits of Applying these preprocessing steps, we have to be careful while choosing the technique.
We verified that a mix of two preprocessing strategies demonstrated great execution. Lemmatization and lowering of words work best in our case. And it highly depends on the data. The paper concludes this study is significant in that it allows better decision-making about which preprocessing type should be
selected according to the purpose of the study or the type of the construction model.

References

1. Diksha Khurana, Aditya Koli, Kiran Khatter. "Natural Language Processing: State of The Art, Current Trends and Challenges".

2. Sethunya R Joseph, Kutlwano Sedimo, Hlomani Hlomani. "Natural Language Processing: A Review".

3. Jindal, Nitin, and Bing Liu. "Opinion spam and analysis." Proceedings of the 2008 International Conference on Web

4. Chapelle O, Schölkopf B, Zien A (2006) Semi-supervised learning. Vol. 2. Cambridge: MIT press.

5. HoSung Woo, JaMee Kim,WonGyu Lee "Validation of Text Data Preprocessing Using a Neural Network Model"

6. Hossein Fani, Masoud Bashari, Fattane Zarrinkalam "Stopword Detection for Streaming Content"

7. Wael Etaiwi, Ghazi Naymat "The Impact of applying Different Preprocessing Steps on Review Spam Detection"

8. Crawford, Michael, Taghi M. Khoshgoftaar, and Joseph D. Prusa. "Reducing feature set explosion to facilitate real world review spam detection." The Twenty-Ninth International Flairs Conference. 2016.

9. Silva, Catarina, and Bernardete Ribeiro. "The importance of stop word removal on recall values in text
categorization." Neural Networks, 2003. Proceedings of the International Joint Conference on. Vol. 3. IEEE, 2003.

10. Klatt, Benjamin, Klaus Krogmann, and Volker Kuttruff. "Developing Stop Word Lists for Natural Language
Program Analysis." Proceedings of WSRE'14 (2014).

11. Moshe Ben-Bassat, Karin L. Klove, and Max H. Weil. 1980. Sensitivity analysis in Bayesian classification models:
Multiplicative deviations. IEEE Trans. Pattern Anal. Mach. Intell. , 3 (1980), 261–266.