Sentiment Analysis on IMDB Movie Reviews

Evaluation and comparison of multi machine learning approaches to predict a binary sentiment of a movie review

Roman Studer
BSc Data Science Student,
FHNW Brugg-Windisch, Switzerland
roman.studer1@students.fhnw.ch

Lukas Gehrig
BSc Data Science Student,
FHNW Brugg-Windisch, Switzerland
lukas.gehrig@students.fhnw.ch

Vincenzo Timmel
BSc Data Science Student,
FHNW Brugg-Windisch, Switzerland
vincenzo.timmel @students.fhnw.ch

***Abstract*— In this report we look at the binary sentiment classification of reviews (Positive or Negative sentiment) with different Machine Learning algorithms. From classical algorithms like linear regression to deep learning models. We explain how the data was prepared and how to interpret the results of the models.**

Keywords—component, formatting, style, styling, insert (*key words*)

I. Introduction

A. Goal of the analysis

This text classification task is focused on the evaluation and comparison of several natural language processing methods for sentiment classification. For this purpose, we use a dataset with 50 thousand film reviews from the IMDB[1] platform. A dataset that is often used for benchmarks in the NLP environment. The dataset is suitable for a binary sentiment classification as it is labeled with the binary feature "sentiment" which either takes on the value "positive", if the sentiment of the review is positive, or "negative" if the sentiment is negative [1]. An example record of the dataset would be:

Table 1, Example of a review in the IMDB dataset

Index	Review	Sentiment
124	“I like this movie”	positive

B. Process

At the start of the process, functions were defined that allow a uniform pre-processing of the data. This ensures that all employed models receive the same data as input. In chapter II we describe the pre-processing in more detail. With the pre-processed data several models were trained and their accuracy on train and test set was recorded. In a further step, the models have been compared. Assumptions about the results, how they were obtained and whether they are robust, are given at the end of the report in chapter 5.

II. Preprocessing

Language offers us humans a way to communicate complex issues in an adaptable and national, even universal, way. This complexity is reflected in the almost endless possibilities of language use. Since machine learning models require numerical values as input, a transformation of the text must take place. In most cases, the individual words of a sentence are divided into a list and each of them is transformed into its basic form. The process of transformation into the basic form is called stemming. Stemming (stem form reduction, normal form reduction) is the term used in information retrieval as well as in linguistic computer science to describe a procedure by which different morphological variants of a word are reduced to their common root, e.g. the declension of “Wortes” to “Wort” and conjugation of "gesehen" or "sah" to "seh". Models like BERT or T5 can also use unpreprocessed texts as input [2]. The following steps were performed on the IMDB Movie Review dataset:

1. remove the HTML tags. These are not relevant for understanding the input.
2. removing brackets around the text
3. removing special characters. E. g. ?,!,/ etc.
4. removing stop words. Stop words are words that contain little information. For example, connection words. They are removed.
5. put words into the basic form. Stemming.

For Transformer models like BERT, steps 4. and 5. are not applied. The target variable Sentiment was also adjusted. An encoding of the binary target variable was made. A positive review is now annotated with the integer 1, a negative review with the integer 0. Table 2 shows an example of the prepared text:

Table 2, Example of a review after preprocessing

Index	Review	Sentiment
15594	[“saw”, “thi”, “movi”, “wa”,..]	0

A. Train-Test-Split

For reproducibility, a function was provided which returns the same split of input data for all models. The 50 thousand reviews are split 80% (40 thousand reviews) into a test set. The training set consists of the remaining 20% (10 thousand reviews).

III. Modelling

A. Baseline Model

As a basic comparison, a baseline model was created. The approach of this model is to count positive and negative words in a review and suggest a positive or negative sentiment based on this count. A list of positive words as well as a list of negative words was imported. Since there is no

[1] IMDB: Internet Movie Database

learning algorithm behind this method, the score was calculated on the test dataset only.

B. Matrix representation

1) Bag of Words

Bag of Words, a method often used for document classification based on the frequency of terms, converts a collection of texts/documents into a matrix with the token frequency as elements. Where the row represents the respective review, and the column represents a word. Grammar and arrangement of input is not included in this model. In the trainset, a total of 6209089 different words were counted after pre-processing. Bag of Words was used as input of a Linear Regression, SVM as well as Multinomial Naive Bayes.

2) TF-IDF

TF-IDF stands for Term Frequency - Inverse Document Frequency and is calculated for each word in the corpus. This value gives an insight into the importance of a term. The Term Frequency, i.e. the relative frequency of a term in a document combined with the relative, logarithmic frequency over all documents (formula 1) gives words that occur frequently and in many documents less weight than words that only occur in a few documents.

$$tf(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}$$

$$\mathrm{idf}(t,D) = \log \frac{N}{|\mathrm{d} \in \mathrm{D} : \mathrm{t} \in \mathrm{d}|}$$

Where TF-IDF is a combination of the two formulas:

$$tf - idf(t,d,D) = tf(t,d) \cdot idf(t,D)$$

The result of TF-IDF is also a matrix with the dimensions $(n\_samples, 6209089)$.

C. Classic models

In addition to the baseline model, classical machine learning models were used at the beginning. These included OLS linear regression with l2 regularization, support vector machines with hinge loss and a multinomial Naive Bayes model. Accuracy was measured for the three models using Bag of Words and TF-IDF as input.

D. Deepl Learning Models

As an extension, two deep learning models were trained. The data used were prepared with the same preprocessing steps as described in chapter II..

1) Bidirectional LSTM

LSTM uses a pre-trained word embedding. Due to memory constraints, the fastText Mini version was used for this case. GloVe (Global Vectors), another word embedding, was also imported for comparison. Thus, two models with different embeddings were trained. The LSTM model was trained with a maximum length of 128, a training batch size of 16, validation batch size of 6 and with 5 epochs.

2) BERT

Another language model, BERT, was trained on a corpus with 800 million words from BooksCorpus and 2,500 million words from Wikipedia. The input data was prepared as in Chapter II, except for steps 4 and 5. A maximum length of 512, a batch size of 16 was used for 4 epochs.
Accuracy was recorded for all models on training and test set. The results are discussed in the following chapter.

IV. Evaluation

The evaluation of the models was performed by the metric Accuracy. Accuracy indicates the relative frequency of a correct prediction. A well-known metric for a binary classification. For learning algorithms such as bidirectional LSTMs or BERT, in addition to the final score, the progression of the metric over the training epochs is interesting and important for detecting overfitting.

Model	**Accuracy**	
	Train	**Test**
Baseline	-	0.7131
Linear Regression (TF-IDF)	0.9963	0.7512
Linear Regression (BoW)	0.9963	0.75
SVM (TF-IDF)	0.9904	0.5829
SVM (BoW)	0.9904	0.5112
Multinomial Naïve Bayes (TF-IDF)	0.9963	0.751
Multinomial Naïve Bayes (BoW)	0.9963	0.7509
LSTM (fastText)	0.99055	0.8358
LSTM (GloVe)	0.9355	0.8739
BERT	0.87663	0.7186

We see a high score for the training set and a significantly lower score for the test set in all models. This could indicate overfitting of the models. Especially for the BERT model, no significant improvement to the baseline was achieved. This is partly due to the fact that the model actually overfitted. Training with a higher batch size, in addition to a high dropout rate could mitigate this. However, due to hardware constraints, this option could not be tested. LSTM provides by far the best result with an improvement of over 16% over the baseline. This with the use of GloVE as embedding. SVM provides a poor result. The large difference to the training score and test score also indicates overfitting here.

References

[1] "Sentiment Analysis." http://ai.stanford.edu/~amaas/data/sentiment/ (accessed Oct. 31, 2021).
[2] "nlp - Using trained BERT Model and Data Preprocessing," *Stack Overflow*. https://stackoverflow.com/questions/63979544/using-trained-bert-model-and-data-preprocessing (accessed Nov. 03, 2021).

Tables