Proceedings of IYSC, (2022), vol. 11

Journal homepage: http://journals.sdu.edu.kz/index.php/iysw

FAST AND RELATIVELY ACCURATE SENTIMENT ANALYSIS FOR THE KAZAKH LANGUAGE

N. Manteyeva

Suleyman Demirel University, Kaskelen, Kazakhstan

Abstract

This paper constructs a fast and accurate sentiment analysis model for the Kazakh language. The main method for text classification is based on TF-IDF-based tokens trained with Logistic Regression. The processing and modeling stages are fully implemented in the PySpark framework. The proposed method has shown an accuracy level of 82% on an evenly distributed test dataset. As a byproduct of the work, we have collected a list of words in the Kazakh language that could signal the negativity/positivity of the given review.

Keywords: sentiment analysis, natural language processing, Kazakh language.

Introduction

Sentiment analysis [1] is a computational study of people's emotions and feelings towards a particular object. A popular problem in sentiment analysis lies in the classification of the text's sentiment as positive or negative. This problem could be viewed as a text classification problem [2]. In general, the problem of sentiment classification is well studied [1] for the English language. There is a wide variety of methods ranging from classical machine learning to deep learning-based methods. However, for certain classes of language, there is a very limited number of methods for sentiment classification. One example of such a language is the Kazakh language. There are some works on sentiment analysis for the Kazakh language [3, 4, 5]. Some of these works are based on rules and heuristics [3, 4], while some [5] are based on resource-hungry LSTM neural network architecture. There is a need for a classical machine learning method with relatively high accuracy and relatively low computational speed. A requirement for low computational speed comes from the needs of business applications. For real-time prediction, the speed of the model plays an important role.

It could be noted that sentiment analysis for texts written in the Kazakh language could be done indirectly by translating the text into the English language first. As there are plenty of sentiment analysis methods available for the English language, the solution seems pretty viable. The downside of the method lies in its financial cost. Solutions for automatic translation charge based on the number of characters/words translated. If one performs a machine translation on a large corpus of text, the financial cost of the process is going to be significantly high. For this reason, there is a need to construct fast and somewhat accurate sentiment analysis methods for the Kazakh language.

Methods

Dataset

The original dataset comes from the Amazon Product Review dataset [1]. The reviews in the given dataset are written in the English language. The dataset contains more than 142 million reviews and takes up more than 20Gb of volume. We have selected reviews from the category of Office products. This category contains more than 53,000 reviews. The reviews are divided into 5 categories according to the number of stars given, so the values range from 1 to 5.

Our process of data collection consisted of two steps:

Choosing a sample of positive and negative reviews.

Using Google Translate to translate the chosen reviews from English to the Kazakh language.

Let's discuss each step.

Firstly, we have selected a sample of positive and negative reviews. Since Google Translate has a limitation on the size of the input text, we could not select all reviews for translation. For this reason, we have selected 1000 positive and 1000 negative reviews. For our purposes, we considered reviews with 1 or 2 stars as negative reviews, while the reviews that had 5 stars were considered positive. We then randomly selected 2000 reviews from the each of two groups.

Secondly, we have used Google Translate to translate the selected reviews into the Kazakh language. To do that we have collected both positive and negative reviews as txt files, and passed these files into Google Translate to translate the given reviews from English to the Kazakh language.

In the end, there are 2000 reviews written in the Kazakh language.

Text Preprocessing

Before proceeding with building a sentiment analysis model, one needs to process the given dataset. There are three types of text processing steps performed:

Removal of digits

Changing the case into lowercase

Building tokenization dictionary

As we wanted to focus on the words only, we have removed characters corresponding to digits from the given texts. Next, all texts were transformed to lowercase. This step reduced the number of distinct words in the dataset. Finally, we have to build a dictionary of tokens and provided each word/token with its corresponding integer index. This step is done to prepare texts

for machine learning analysis.

Feature Engineering

TF-IDF terms

TF-IDF is a popular feature construction technique used for text data [6]. Term frequency-inverse document frequency (TF-IDF) is a feature vectorization method widely used in text mining to reflect the importance of a term to a document in the corpus. Denote a term by t, a document by d, and the corpus by D. Term frequency TF(t,d) is the number of times that term t appears in document d, while document frequency DF(t,D) is the number of documents that contains term t. If we only use term frequency to measure the importance, it is very easy to over-emphasize terms that appear very often but carry little information about the document, e.g., "a", "the", and "of". If a term appears very often across the corpus, it means it doesn't carry special information about a particular document. Inverse document frequency is a numerical measure of how much information a term provides:

$$IDF(t, D) = (log|D| + 1) \,/\, (DF(t, D) + 1)$$

where |D| is the total number of documents in the corpus. Since logarithm is used, if a term appears in all documents, its IDF value becomes 0. Note that a smoothing term is applied to avoid dividing by zero for terms outside the corpus. The TF-IDF measure is simply the product of TF and IDF

$$TFIDF(t, d, D) = TF(t, d) \cdot IDF(t, D)$$

There are several variants of the definition of term frequency and document frequency. In spark.mllib, we separate TF and IDF to make them flexible. It is worth noting that PySpark framework uses hashing technique to speed up the calculation of TF-IDF terms.

Selecting Significant Features

After the construction of TF-IDF tokens, there were 212 features in total. Considering the size of our dataset, there was an excess of features to work with. In order to reduce the number of features, we have performed a chi-square test [7] to leave only those features that showed statistical significance with respect to the target variable. After using the chi-square test we selected 52 features that showed statistical significance.

Feature Scaling

In general it is a good practice in machine learning to normalize features before applying any machine learning algorithm. In our case, scaling had a second benefit of allowing us to identify

important features. The feature importance provides a measure of the influence of the given feature on the target variable. In the case of Logistic Regression, feature importance could be measured by the absolute value of the corresponding weight coefficient. For the validity of this approach, the features need to be scaled.

We have used a min-max feature scaler. It works as follows

$$x' = (x - min(x)) / (max(x) - min(x))$$

After using this scaling technique all features range within [0, 1] interval.

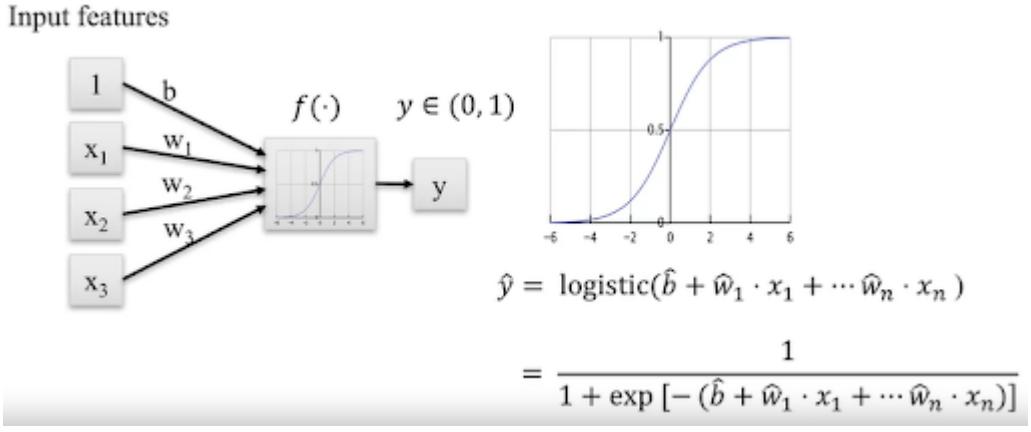
Machine Learning Modeling

Train/Test Data Split

As part of a standard technique for machine learning modeling we randomly split the dataset into training and testing datasets. The testing dataset comprised 20% of the original dataset. After we have trained the machine learning model on the training dataset, we could safely test its performance on the test dataset. As for the implementation in PySpark we have randomSplit() with input parameters of [0.8, 0.2].

Logistic Regression

Logistic regression is a popular and widely used machine learning model for classification problems.



Logistic regression offers the following advantages over rule-based heuristics

Statistical nature

Higher accuracy

Logistic regression offers the following advantages over neural network-based models:

Lower training and prediction speeds

Higher interpretability

PySpark offers its own implementation of logistic regression available as pyspark.ml. classification.LogisticRegression class. We have used this class for modeling our dataset.

Results

F1-Scores

We have computed the F1-score for all three modeling methods: Heuristics, Logistic Regression and LSTM. The results are given below.

Model	F1-score
Heuristics-based model	0.61
Logistic Regression-based model	0.81
LSTM based model	0.89

As we can see Logistic Regression-based model showed significantly higher results compared to the heuristics-based model

Train/Predict times

We have computed training and prediction times, in seconds, for all three modeling methods.

Model	Train Time (sec)	Predict Time (sec)
Heuristics based model	n/a	0.06
Logistic Regression based model	0.72	0.14
LSTM based model	24.3	0.85

As we can see the Logistic Regression based method showed significantly lower training and prediction times compared to the LSTM based approach.

Trigger Words

By calculating feature importance for the Logistic Regression model we could identify a list of words that highly affect the sentiment of the review. By using this list of words, it is possible to come up with a rule-based algorithm that judges the sentiment of a review by the inclusion of these highly influential/trigger words.

Conclusion

We have developed a sentiment analysis of review messages in the Kazakh language. We have used Logistic Regression to model the data, and PySpark as a computing environment for our experiments. Our study has shown that a Logistic Regression-based model could be significantly more accurate than heuristics-based models. Moreover, the Logistic Regression based model showed significantly faster train/prediction times compares to the LSTM based model. As a byproduct of our research, we have collected a list of trigger words that influence the final class of review messages the most.

References

1. Medhat, W., Hassan, A., & Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. *Ain Shams engineering journal*, *5*(4), 1093-1113.
2. Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., & Brown, D. (2019). Text classification algorithms: A survey. *Information*, *10*(4), 150.
3. Yergesh, B., Bekmanova, G., & Sharipbay, A. (2017, October). Sentiment analysis on the hotel reviews in the Kazakh language. In *2017 International Conference on Computer Science and Engineering (UBMK)* (pp. 790-794). IEEE.
4. Yergesh, B., Bekmanova, G., & Sharipbay, A. (2019, January). Sentiment analysis of Kazakh text and their polarity. In *Web Intelligence* (Vol. 17, No. 1, pp. 9-15). IOS Press.
5. Sakenovich, N. S., & Zharmagambetov, A. S. (2016, September). On one approach of solving sentiment analysis task for Kazakh and Russian languages using deep learning. In *International Conference on Computational Collective Intelligence* (pp. 537-545). Springer, Cham.
6. https://spark.apache.org/docs/latest/mllib-feature-extraction.html
7. https://en.wikipedia.org/wiki/Chi-squared_test