ISTANBUL TECHNICAL UNIVERSITY

MSc. Big Data and Business Analytics

FINAL PROJECT REPORT

TEXT CLASSIFICATION ON
AMAZON CUSTOMER REVIEWS DATASET

by

Ece Şimşek

13 July, 2021

TEXT CLASSIFICATION ON AMAZON CUSTOMER REVIEWS DATASET

FINAL PROJECT REPORT

by
Ece Simsek

Abstract

At the present time, the spread of e-commerce and the increase in product diversity have created a need for customers to choose the right and most suitable product. Therefore, when customers want to choose the most suitable product for themselves and their budget, the reviews of other customers about their experiences with that product have become more valuable. For this reason, it is possible to say that customer reviews provide an objective feedback to the customer who will buy the product in his purchasing process.

While customers make comments about the product, they also make different types of ratings to summarize their reviews about the product. These ratings can be identified by the number of stars or a different numerical rating type. These summary ratings may vary depending on the customer's character, mood or personal differences of opinion and sometimes do not fully reflect the experience of the product. At this point, the main thing that reflects the relevant feedback is actually the customer review text itself. In this context, the main purpose of this project is to analyze the reviews made by customers using NLP techniques and try to predict the most accurate rating based on the meaning of the text.

Keywords

Text Mining, NLP, Machine Learning, Supervised Learning, Text Classification, Ensemble Learning

1.Dataset

Customer reviews collected from the Amazon e-commerce website are aggregated from all reviews by each customer, with each customer review representing an integer ranging from one star to five stars. Customer reviews collected from the Amazon e-commerce website are aggregated from all reviews by each customer, with each customer review representing an integer ranging from one star to five stars. Therefore, we will need a supervised and multiclass classifier in this study. Besides, we will try to perform feature extraction to apply the best machine learning algorithm to solve our classification problem using Natural Language Processing (NLP) techniques like word embedding, topic modeling, and dimension reduction etc.

The entire dataset collected from Amazon contains product reviews and metadata, including 233.1 million reviews spanning May 1996 - Oct 2018. There are 29 categories in total, but the category "All beauty" was chosen within the scope of this study. It contains a total of 371,345 customer reviews, has 12 features and the size of the dataset is 184 MB in total. These features and their explanations are briefly as follows:

- **reviewerID -** ID of the reviewer
- **asin -** ID of the product
- **reviewerName -** name of the reviewer
- **vote -** helpful votes of the review
- **style -** a dictionary of the product metadata
- **reviewText -** text of the review
- **overall -** rating of the product
- **summary -** summary of the review
- **unixReviewTime -** time of the review (unix time)
- **reviewTime -** time of the review (raw)
- **image -** images that users post after they have received the product
- **verified -** information whether the customer who made the review purchased the product or not

2. Literature Review

As a first example, it aims to apply supervised learning algorithms to predict a review's rating on a given numerical scale based on text. There some different ML algorithms were tried such as Naive Bayes, Perceptron, and Multiclass SVM, then compared predictions with actual ratings. For the preprocessing stage, they also followed up various feature selection algorithms such as using an existing sentiment dictionary, building our own feature set, removing stop words, and stemming.

Several supervised models are built. It includes both traditional algorithms such as naive bayes, linear support vector machines, K-nearest neighbor, and deep learning metrics such as Recurrent Neural Networks and convolutional neural networks. Comparing the accuracy of these models gives a better understanding of the polarized attitudes towards the products..

It is stated that there are two different methods to perform sentiment analysis such as the Lexicon-based method and Machine Learning method. In lexicon-based sentiment analysis, it is calculated from the semantic orientation of words or phrases present in a text. However, in ml methods, it requires annotated datasets. It is critical that sentiment analysis is highly domain-oriented and centric because the model developed for one domain like a movie or restaurant will not work for the other domains like travel, news, education, and others. If the algorithm has been trained with the fashion data and is used to predict food and travel-related sentiments, it will predict poorly.

3.Data Preparation

3.1. Data Cleaning

All stages of the project were written in python programming language and python libraries were used in all analyzes and all machine learning algorithms used. The df is read from the Amazon Reviews Dataset. The file can be

downloaded from the website and installed from the local file, or the file is accessed and read directly from the repository.

As can be seen from the pandas output below, the overall field is stored as float, unixReviewTime stored as integer, verified stored as boolean and the rest of the fields are stored as strings (objects).

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 371345 entries, 0 to 371344
Data columns (total 12 columns):
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   overall         371345 non-null  float64
 1   verified        371345 non-null  bool   
 2   reviewTime      371345 non-null  object 
 3   reviewerID      371345 non-null  object 
 4   asin            371345 non-null  object 
 5   reviewerName    371307 non-null  object 
 6   reviewText      370946 non-null  object 
 7   summary         371139 non-null  object 
 8   unixReviewTime  371345 non-null  int64  
 9   vote            51899 non-null   object 
 10  style           125958 non-null  object 
 11  image           8391 non-null    object 
dtypes: bool(1), float64(1), int64(1), object(9)
```

Figure 1. Non-Null Counts and Data Types

On the other hand, as can be seen below, 399 lines were found to be "na" in the reviewText column. Therefore, first of all, these empty lines have been dropped. After that, due to mistakes made during data collection, there is duplicated data in the reviewText field. In the next step, these duplicated rows are removed from the dataset. Thus, the data set consisting of 371,345 rows decreased to 319,357 rows after the basic data cleaning process.

```
df.isna().sum()
```

```
overall               0
verified              0
reviewTime            0
reviewerID            0
asin                  0
reviewerName         38
reviewText          399
summary             206
unixReviewTime        0
vote               319446
style              245387
image              362954
dtype: int64
```

Figure 2. Null Counts of Fields

In the final stage of data cleaning, the unixReviewTime is converted from Unix time to classical datetime format and after that reviewTime column is dropped from dataframe.

3.2. NLP Pre-Processing

The final dataframe for the model will be drawn from the “reviewText” column.The “overall” column will serve as the labels for the text classification process.

Within the scope of the project, the reviewText field will be used to create the final dataframe of the model. The goal here is to generate tokens for each review. Tokens extracted from these reviews will create a corpus for the vocabulary to be created later.

The original version of a sample review can be seen below. In the next sections, the final versions after the lemmatization, removing of stop word, punctuations, digits and also tokenization stages will also be stated.

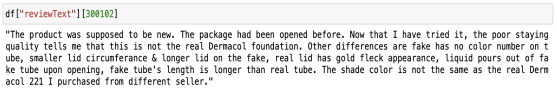

Figure 3. Sample Review Text

3.2.1.Lemmatization

Extracting the root of a word is an important step for creating the ultimate vocabulary. So, in this section, it is aimed to extract the root of a word using semantic analysis rather than a rule based algorithm. Therefore, lemmatization was used instead of stemming.

WordNetLemmatizer() from the NaturalLanguageToolkit (NLTK) library is used for the lemmatization process. It applies to each word but it is dependent on sentence structure to understand the entire context. Therefore there is a need to have part-of-speech tags associated with each word. According to the mentioned part-of-speechs, a function named lemmatize_word has been created to perform the lemmatization process. Afterwards, the lemmatize_rev function is applied to update the rooted words in the related reviewText field.

The lemmatization process for this dataset took 12 minutes and 39 seconds. In order not to wait for this time in each run, the data frame with the rooted words in the reviewText column was saved to the local in csv format. The output after the lemmatization process is as follows:

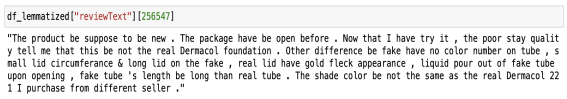

Figure 4. Sample Review Text After NLP Operations

3.2.2. Removing Stop Words

Stop words are the most commonly used words that include pronouns (e.g. us, she, it, they), articles (e.g. the, of), and prepositions (e.g. under, from, off). These words do not help to distinguish one document from another and are therefore generally dropped in the NLP preprocessing stage. To remove stop words, the stop words list of the NLTK library prepared for the English language was used. No extra words have been added to this list.

3.2.3. Removing Punctuations & Digits

After removing stop words, the preprocessed reviews are cleaned by dropping punctuations. Only whitespaces and alphanumeric characters are kept using regular expressions. After that, digits are also removed using regular expression.

3.2.4.Removing Extra Spaces

After removing the punctuation marks, it was seen that there were reviews with more than one space between words in the data set. Therefore, the preprocessed reviews are also cleaned by dropping extra spaces.

3.2.5. Converting Lower Case

In the final stage of normalization, every letter is converted to lower case to standardize the reviews. Thus, the words "beauty" and "Beauty" will not be considered separate words in a vocabulary/corpus.

3.2.6. Tokenization

In order to create a corpus in a vocabulary, each document (review) is broken down into individual words or tokens. This process is called tokenization. No library was used for this process, and since we removed the extra spaces before, the document was split from a single space (" ").

After all NLP preprocessing steps, the sample output is as follows:

```
['product', 'suppose', 'new', 'package', 'open', 'try', 'poor', 'stay', 'quality', 'tell', 'real', 'dermacol', 'fou
ndation', 'difference', 'fake', 'color', 'number', 'tube', 'small', 'lid', 'circumferance', 'long', 'lid', 'fake',
'real', 'lid', 'gold', 'fleck', 'appearance', 'liquid', 'pour', 'fake', 'tube', 'upon', 'opening', 'fake', 'tube',
'length', 'long', 'real', 'tube', 'shade', 'color', 'real', 'dermacol', '221', 'purchase', 'different', 'seller']
```

Figure 5. Sample Output of Tokenization

3.3. Creating Vocabulary

Vocabulary created using the Gensim library is the key-value pairs of all unique tokens from each product review. As can be seen in the output below, a lookup ID is assigned to each unique token. This vocabulary will be used in applications such as Bag of Words, TF-IDF and word embedding, which will be indicated in the next sections.

```
vocabulary.token2id
```

```
{'great': 0,
 'addition': 1,
 'baseball': 2,
 'book': 3,
 'haveinformation': 4,
 'husband': 5,
 'library': 6,
 'n': 7,
 'negro': 8,
 'read': 9,
 'start': 10,
 't': 11,
 'tthank': 12,
 'want': 13,
 'aspect': 14,
 'cover': 15,
 'game': 16,
 'informative': 17,
 'already': 18,
```

Figure 6. Vocabulary of Words

4. Count-Based Feature Engineering

For a machine learning model to work with text input, firstly the document must be vectorized. This simply means that the input must be converted to numeric values. In this scope, there are both count-based approaches and word embedding approaches that also reduce the number of dimensions.

4.1. Bag of Words (BOW)

The most common approach in expressing text as a set of features is getting the token frequency. Each entry to the dataframe is a document while each column corresponds to every unique token in the entire corpora. The row will identify how many times a word appears in the document. The bow model for the sample review is below:

```
Word: color, Freq: 2
Word: number, Freq: 1
Word: tell, Freq: 1
Word: product, Freq: 1
Word: small, Freq: 1
Word: seller, Freq: 1
Word: purchase, Freq: 1
Word: quality, Freq: 1
Word: appearance, Freq: 1
Word: long, Freq: 2
Word: open, Freq: 1
Word: foundation, Freq: 1
Word: stay, Freq: 1
Word: try, Freq: 1
Word: package, Freq: 1
Word: opening, Freq: 1
Word: different, Freq: 1
Word: new, Freq: 1
Word: poor, Freq: 1
Word: suppose, Freq: 1
Word: real, Freq: 4
Word: difference, Freq: 1
Word: lid, Freq: 3
```

Figure 7. Sample Output of Bag of Words Model

4.2. TF - IDF

Term Frequency-Inverse Document Frequency (TF-IDF) is another approach where continuous values are assigned to tokens.

The term frequency is a raw count of instances a word appears in a document. *The inverse*

document frequency means, how common or rare a word is in the entire document set. The closer it is to 0, the more common a word is. So, if the word is very common and appears in many documents, this number will approach 0. Otherwise, it will approach 1.

Multiplying these two numbers results in the TF-IDF score of a word in a document. The higher the score, the more relevant that word is in that particular document.

In order to determine the weights in the TF - IDF model created using the Gensim library, the BOW count-based feature engineering technique that we created before was used. The output of the TF-IDF model is as follows:

```
Word: color, Weight: 0.105
Word: number, Weight: 0.107
Word: tell, Weight: 0.077
Word: product, Weight: 0.028
Word: small, Weight: 0.059
Word: seller, Weight: 0.088
Word: purchase, Weight: 0.056
Word: quality, Weight: 0.059
Word: appearance, Weight: 0.113
Word: long, Weight: 0.104
Word: open, Weight: 0.082
Word: foundation, Weight: 0.098
Word: stay, Weight: 0.071
Word: try, Weight: 0.048
Word: package, Weight: 0.081
Word: opening, Weight: 0.130
Word: different, Weight: 0.069
Word: new, Weight: 0.068
Word: poor, Weight: 0.105
Word: suppose, Weight: 0.091
Word: real, Weight: 0.343
Word: difference, Weight: 0.075
Word: lid, Weight: 0.314
```

Figure 8. Sample Output of TF-IDF Model

5. Word Embedding for Feature Engineering

Count-based techniques do not give regard to word sequence and sentence structure, and thus lose semantics. The Word2Vec technique from Gensim lib actually embeds meaning in vectors by quantifying how often a word appears within the vicinity of a given set of other words. Since the default parameters are not changed while using this algorithm, the vector size is 100, the window size is 5, and the alpha value is 0.025.The output of the word2vec model is as follows:

	0	1	2	3	4	5	6	7	8	9	...	90	91	92	
great	-1.201949	0.144700	0.574869	2.752909	-0.388276	-1.574940	0.477391	0.049548	-1.164826	-0.106699	...	1.994949	2.316634	1.655789	1.26
husband	0.044235	-0.624490	0.766864	-0.096014	-1.737497	-1.962552	1.309048	2.582413	-0.380935	0.499077	...	-0.424558	0.666695	0.228896	-0.98
want	-1.975531	1.179926	-1.058031	0.623056	-0.554303	1.691741	-0.853700	0.850673	0.688437	-1.651572	...	-0.635158	0.537367	-0.549163	-1.61
read	-0.597246	-2.358415	0.198330	-0.574358	-0.256990	0.211905	0.574853	1.306706	-0.310360	-1.215431	...	0.877756	2.454601	-0.734411	-2.04
negro	0.112753	0.003688	-0.110379	0.157676	0.004753	0.150588	0.081015	0.107179	0.054396	0.117308	...	-0.184214	-0.046896	-0.078399	0.09

5 rows × 100 columns

Figure 9. Sample Output of Word2Vec

After applying the word2vec algorithm, the average of the vector values on the basis of the document was taken to reach the final data frame. Thus, the final dataframe was created for modeling, with the columns representing the dimensions and the rows representing the document.

6. Exploratory Data Analysis

In order to better understand the “All Beauty” category from Amazon Customer Reviews dataset, some data analysis was conducted on the dataset by using Latent Dirichlet Allocation (LDA) technique for Topic Modelling besides Matplotlib for visualization. In addition, an analysis was carried out on word similarities with the word2vec algorithm applied in the previous “Feature Engineering” section.

6.1. LDA and Topic Modelling

"LDA is a generative probabilistic model of a corpus. The basic idea is that documents are represented as random mixtures over latent topics, where each topic is characterized by a distribution over words." In other words, it is an unsupervised learning method that clusters documents together according to topic.

In the scope of the project, LDA technique with bag of words was applied for topic modelling. In this context, product reviews will have weights assigned to each of the topics and the topics themselves will have weights on every token.

A sample of topics created using the LDA method and the top 5 words related to this topic are as follows:

```
Topic 4:
 brush 0.025476820766925812
 razor 0.024815315380692482
 shave 0.024490581825375557
 use 0.014448480680584908
 shaver 0.013407531194388866
```

Figure 10. Sample Output of Topic Modelling

6.2. Word2Vec Word Similarities

As mentioned in the previous section, the Word2Vec technique measures how often a word appears near a given set of other words, placing meaning into vectors.

The output showing the top 5 words similar to a selected word, made using this technique, is as follows:

```
word_vec.wv.most_similar('shave', topn=5)
```

```
[('shaving', 0.8168691992759705),
 ('shaves', 0.735748291015625),
 ('razor', 0.6707531213760376),
 ('shaver', 0.5992351770401001),
 ('blade', 0.5751073360443115)]
```

Figure 11. Sample Output of Word2Vec Word Similarities

Another analysis output using the same technique is as follows:

```
blade: ['blades' 'razor' 'feathers' 'cartridge' 'mach']
skin: ['face' 'complexion' 'flaky' 'redness' 'rosacea']
hair: ['wavy' 'frizzy' 'curly' 'curl' 'strand']
shampoo: ['conditioner' 'detangler' 'poo' 'pantene' 'dandruff']
eye: ['eyes' 'eyelid' 'undereye' 'crease' 'wrinkle']
parfume: ['perry' 'prada' 'fragrence' 'flavors' 'amor']
lip: ['lips' 'chapstick' 'lipstick' 'chap' 'lipgloss']
fruit: ['cinnamon' 'rosemary' 'grape' 'chamomile' 'melon']
pencil: ['liner' 'sharpener' 'eyeliner' 'eyeshadow' 'palette']
wave: ['curl' 'curls' 'bouncy' 'wavy' 'frizz']
teeth: ['tooth' 'gum' 'molar' 'tray' 'mouth']
```

Figure 12. Sample Output of Word2Vec

6.3. Word Cloud Visualizations

The most common words according to each overall rating are drawn as word clouds below, in order to see the context of the reviews:

Figure 13. WordCloud for 1 Star Rating

Figure 14. WordCloud for 2 Star Rating

Figure 15. WordCloud for 3 Star Rating

Figure 16. WordCloud for 4 Star Rating

Figure 17. WordCloud for 5 Star Rating

7. Modelling

4 different classical machine learning algorithms which were Random Forest, Decision Tree, Extra Tree Classifier and Xgboost were applied to the dataset in order to predict the rate of customers. (rate prediction from 1 to 5). In addition to classical machine learning algorithms, CNN and LSTM deep learning techniques were also used. The outputs of classical machine learning models are as follows:

	Random Forest	Decision Tree	ExtraTree Classifier	XGBoost
Train Accuracy	98.8%	98.6%	99.07%	74.15%
Test Accuracy	64.04%	50.6%	64.47%	67.4%

Table 1. Model Results for Classical Machine Learning Algorithms

As can be seen in the table above, the algorithm that gave the highest accuracy score in the test dataset was Xgboost with 67.4%. However, when the train set accuracy scores are examined, it can be seen that the test sets are much higher than the accuracy scores. The reason for this is the overfit problem during the training stage of the model.

On the other hand, when precision and recall values of each of the classes were evaluated, it was found that it did not predict the 2nd, 3rd and 4th classes very well. The main reason for this is the unbalancing problem in the dataset. In the next section, information about the solution of the unbalancing problem in the dataset will be given.

In addition to classic machine learning algorithms, both LSTM and LSTM + CNN deep learning algorithms were applied on the

dataset. The accuracy and loss metrics on test set of these models are as follows:

	LSTM	LSTM + CNN
Test Accuracy	69.6%	66.7%
Test Loss	0.826	1.13

Table 2. Model Results for Deep Learning Algorithms

As can be seen in the table above, the model applied using only the LSTM algorithm without CNN was the algorithm that gave the best output with the accuracy score of 69.6% and the loss value of 0.83.

7.1.Dealing with Imbalanced Dataset

As can be seen in the bar chart below, customers tend to give 5 stars generally, while they tend to give 2 points rarely. This leads to the imbalanced problem in the dataset.

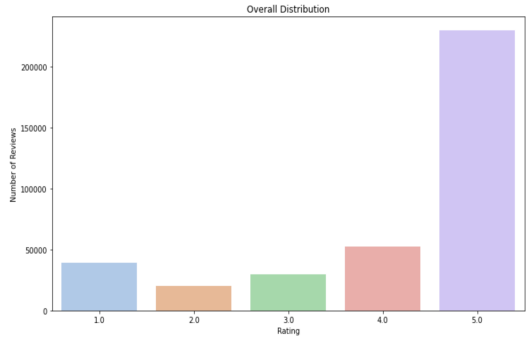

Figure 18. Distribution of Star Ratings

There are many different techniques such as oversampling, resampling, undersampling and smoothing to solve this imbalanced problem in data sets. Many of these techniques can be applied manually as well as various libraries can be used. In this project, the undersampling method was used despite the risk of reducing the size of the data set. Within the scope of this technique, the number of reviews in the 1,3,4 and 5 classes is equalized to the minimum number of reviews with 2 stars which is 19000 rows. Thus, the total number of rows decreased to 95000 and the size of the data set was reduced by 2/3. The distribution of rates after the undersampling method is as follows:

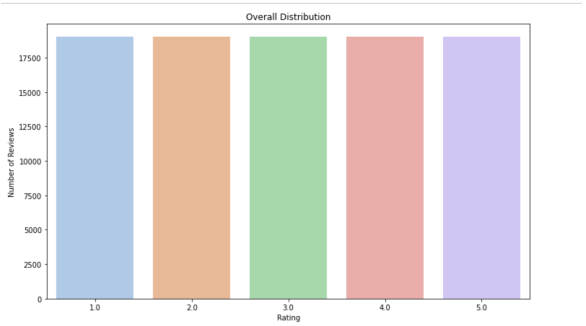

Figure 19. Distribution of Star Ratings after Undersampling

XGBoost Classifier, the classical machine learning algorithm that works best on the balanced data after the undersampling method, has been applied and the confusion matrix with precision and recall values on test set is as follows:

```
Confusion Matrix:

[[2197  885  357  174  187]
 [1070 1200  829  453  248]
 [ 518  911 1135  857  379]
 [ 198  396  666 1464 1076]
 [ 173  207  230  784 2406]]

Accuracy:  0.4422105263157895

Classification report:

              precision    recall  f1-score   support

         1.0       0.53      0.58      0.55      3800
         2.0       0.33      0.32      0.32      3800
         3.0       0.35      0.30      0.32      3800
         4.0       0.39      0.39      0.39      3800
         5.0       0.56      0.63      0.59      3800

    accuracy                           0.44     19000
   macro avg       0.43      0.44      0.44     19000
weighted avg       0.43      0.44      0.44     19000
```

Figure 20. Confusion Matrix and Model Metrics for XGBoost

As can be seen in the figure above, although the overall accuracy has decreased, it has been observed that the model better predicts the 2nd, 3rd and 4th classes. In future studies, model outputs can be evaluated by trying various sampling methods to get more meaningful accuracy scores.

Conclusion

It is an undeniable fact that the importance of the effect of online reviews on customer purchasing behavior is increasing day by day. In this study, customer comments were analyzed using NLP techniques, and rate prediction models were applied over the semantic outputs of these analysis results by using multiclass and binary text classification methods.

Among the models implemented using classical machine learning algorithms, the highest overall accuracy score of 67.4% was achieved using the XGBoost Classifier algorithm for the multiclass classification problem. On the other hand, the model applied using only the LSTM algorithm without CNN was the algorithm that gave the best output with the accuracy score of 69.6%.

On the other hand, due to the imbalanced problem in the dataset, models could not predict the 2.3. and 4th classes enough. For this reason, the undersampling method was used. Although overall accuracy was lower with the balanced dataset, the probability of the models to predict the minority class has increased.

References

[1] http://cs229.stanford.edu/proj2014/Yun%20Xu,%20Xinhui%20Wu,%20Qinxia%20Wang,%20Sentiment%20Analysis%20of%20Yelp's%20Ratings%20Based%20on%20Text%20Reviews.pdf

[2] http://cs229.stanford.edu/proj2018/report/122.pdf

[3] https://pub.towardsai.net/sentiment-analysis-opinion-mining-with-python-nlp-tutorial-d1f173ca4e3c

[4] https://matheo.uliege.be/bitstream/2268.2/2707/4/Memoire_MarieMartin_s112740.pdf

[5] http://jmcauley.ucsd.edu/cse190/projects/fa15/019.pdf

[6] louiefb/amazon-reviews-nlp: Classifying Amazon reviews based on customer ratings using Natural Language Processing

[7] https://github.com/joshivaibhav/AmazonCustomerReview/blob/master/Term%20Project%20Final%20Report.pdf

[8] http://www.ir.juit.ac.in:8080/jspui/bitstream/123456789/22817/1/Amazon%20Rating%20Analysis%20and%20Prediction.pdf