Amazon reviews: comparison between text classification models and topic modeling with different text representation techniques

Serino Antonio

Dipartimento di Informatica Sistemistica e Comunicazione, DISCO , Università degli Studi di Milano Bicocca
Piazza dell'Ateneo Nuovo, 1, 20126 Milano MI

Abstract: The proposed project deals with using a dataset composed of a set of reviews of products purchased on the well-known Amazon e-commerce in order to be able to explore state-of-the-art topics: in particular, two different text representations have been used in order to develop different models of text classification and topic modelling, offering a final starting point as a hypothetical future development.

Key words: Text Mining, Text Representation, Text Classification, Topic Modelling

Contents

1 Introduction

Text mining is an interdisciplinary discipline that deals with the extraction of meaningful information from large amounts of unstructured text. It uses natural language processing, statistical and machine learning techniques to analyse and understand text data. In the modern world, text mining has become an important tool to help companies understand customer sentiments, monitor online reputations, analyse social media data and much more. In addition, text mining has become an important element in academic research, healthcare, public relations, national security and many other industries that need to analyse large amounts of text data.

2 Preprocessing

2.1 Dataset description

The dataset chosen to implement the planned tasks is "Amazon Reviews for Sentiment Analysis": this is already partitioned into two portions, one prepared for training models (3,600,000 reviews) and the other prepared for testing models (400,000 reviews). Each of these reviews is associated with a binary class, 0 or 1, where 0 stands for a negative rating of the review and 1 stands for a positive one.

At first, it was thought that the original partition could be maintained, using the entire dataset to develop the work, but after several attempts, the compu-

tational time for training the models turned out to be too long: it was decided to work on a sampled version of the original train dataset, resulting in a training set of 50,400 reviews and a test set consisting of 21,600 reviews.

2.2 Preprocessing

The data pre-processing starts with the extraction of texts and labels from the original files, compressed in bz2 file format, and inserting them into two lists, composed by the reviews and their respective classification labels.

Next, the ”clean_texts()” function is used, to which the set of reviews is passed: this takes care of removing, for each review, non-significant characters, tokenizing the sentences and removing the stopwords for each of them, returning the list of cleaned reviews as output.

```
[ ]  def clean_texts(texts):
        stwords = stopwords.words('english')
        l = len(texts)/10
        temp_texts = []
        for i in range(len(texts)):
            text = re.sub('\d','0',texts[i])
            if 'www.' in text or 'http:' in text or 'https:' in text or '.com' in text: # remove links and urls
                text = re.sub(r"([^ ]+(?<=\.[a-z]{3}))", " ", text)

            text = re.sub('[^a-zA-Z]', ' ', text)
            text = text.lower()
            text = text.split() #tokenization
            text = [word for word in text if not word in stwords]
            text = ' '.join(text)
            temp_texts.append(text)
            if i%l==0:
                print('--'+str(int(i/l)*10)+'%', end='')
        print('Finishd all')
        return temp_texts
```

Figure 1: *clean_text() function*

2.3 Exploratory analysis

The analysis consists of counting how many elements belonging to the different evaluation classes we have within the train and test partitions of the sampled dataset.

Both partitions consist of perfectly balanced classes:

- within the train dataset we find 25,200 observations belonging to each of the two classes;

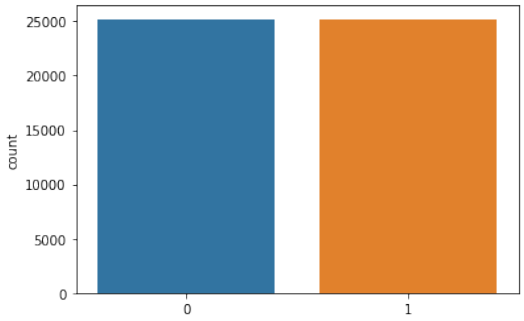

Figure 2: *counting train labels*

- within the test dataset, on the other hand, we find 10,800 observations belonging to each.

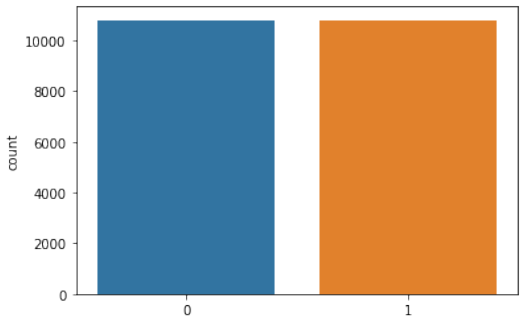

Figure 3: *counting test labels*

3 Text Representation

This section of the document describes the two text representation models adopted to train the models.

3.1 TF-IDF: terms frequency inverse document frequency

TF-IDF is a word importance metric that is used to represent the relevance of a word in a document with respect to a set of documents: the metric is calculated as the product of two components, the frequency of a word within a single document (in our case within a single review) and the inverse of the document frequency, i.e. the unique importance of a word with respect to all documents.

$$W(x,y) = tf(x,y) \cdot \log(\frac{N}{df(x)}) \quad (1)$$

In practice, in order to implement the TF-IDF matrix of reviews, the TfidfVectorizer() method of the sickit learn library has been used: once the vectorizer object

has been instantiated, the set of our reviews is passed to it and it will return a **n*m** matrix in output where **n** is the number of reviews and **m** is the number of terms (in my specific case m will be 500 because it is the maximum value I specified). The adjacency between **n** and **m** will be the weight that each word **m** takes in review **n**.

3.2 Doc2Vec

Doc2Vec is an embedding model. Embeddings are an information representation technique that consists of associating each element of a set of objects with a vector of numbers. In this case, each document is represented as a vector of numbers and these vectors are used to perform various natural language processing tasks.

Doc2Vec is based on an artificial neural network architecture that allows documents to be represented as vectors of numbers. This means that each document is mapped into a vector of fixed size representing its semantic and lexical features. These vectors can then be used to calculate similarity between documents, classify documents into categories or perform other natural language processing tasks. The model is trained on large amounts of text and improves its ability to represent the meaning of documents as the number of training data increases.

In practice, in order to implement Doc2Vec, I first used the **TaggedDocument()** method, which takes care of associating a unique tag to each review (both test and train, since only one portion was created), and then the **doc2vec()** method, to which I passed the tagged reviews on the basis of which the model will be trained, the size of the vector that must represent each review (100), the context window used to define the relationships between words within a review, the minimum number of times a word must be present in documents to be considered (2) and the number of workers (threads) to be used during the training process.

Finally, the trained model was used to create the text representation for the set of train and test reviews.

4 Text classification

Come to this point, I had 2 different representations of the texts, so the idea was to compare them by realising 4 different text classification models: a logistic regression, a support vector machine, a random forest and a neural network.

Comparisons between the results of the realised models are given below.

4.1 TF-IDF

Classifier	Accuracy	Precision	Recall	F-Measure
Logistic Regression	0.833	0.834	**0.832**	0.833
SVM	**0.834**	**0.838**	0.829	**0.834**
Random Forest	0.802	0.812	0.787	0.799
Feed Forward Neural Network	0.822	0.819	0.827	0.823

Table 1: *comparison of text classification models obtained with TF-IDF text representation*

Of the models built for text representation with tf-idf, the **Support Vector Machine** performs best with an **accuracy** of **0.834**, a **precision** of **0.838** and a **F-measure** of **0.834**.
It is only fair to point out that the best performance for **recall** is achieved by **Logistic Regression (0.832)**, but as the difference with that of the SVM (0.829) is small and all other performances are in its favour, it is preferred as the best model.

4.2 Doc2Vec

Classifier	Accuracy	Precision	Recall	F-Measure
Logistic Regression	0.794	0.790	0.799	0.795
SVM	0794	0790	0.802	0.796
Random Forest	0.764	0.769	0.753	0.761
Feed Forward Neural Network	**0.802**	**0.801**	**0.803**	**0.802**

Table 2: *comparison of text classification models obtained with Doc2Vec text representation*

For text representation by Doc2Vec, the best performing model was the **feed forward neural network**: it scored an **accuracy** of **0.802**, a **precision** of **0.801**, a **recall** of **0.803** and an **F-measure** of **0.802**.

4.3 Feed-Forward Neural Network

The architecture of the neural network used is explained below: it is a **feed-forward neural network** implemented using the Keras library. The feed-

forward neural network uses fully connected layers to process data. The structure of the model consists of:

- An input layer receiving 100 attributes, with a ReLU activation function;
- Hidden layers with ReLU activation function and Dropout to prevent overfitting;
- An output layer with one unit and sigmoid activation function.

The model is then filled with the loss function 'binary_crossentropy' and the optimiser 'adam'.

4.4 Output of the text classification phase

Arrived to this point, I had at my disposal two different best performing classifiers for each text representation: **the idea was to construct two datasets with test reviews, whose classification labels will be calculated by the respective best classifier**.

5 Topic modelling

Topic modelling is a text analysis technique capable of identifying topics in a set of documents.
This involves using both supervised and unsupervised machine learning algorithms to analyse the content of documents and group similar words and phrases into distinct topics.
Topics are generally represented as groups of words that frequently appear together and are described by a list of keywords.
My analysis with topic modelling is based on the idea of dividing each dataset constructed using the results of the text classification into two sub-portions, the portion of reviews classified as positive and the portion of reviews classified as negative, in order to be able to compare the topics generated using the coherence metric.

5.1 LDA

The model used is the Latent Dirichlet Allocation of the Gensim library: this is based on the idea that each document in a corpus is a combination of different topics and that each topic is represented by a set of words.
In practice, the algorithm simply assigns a probability of belonging for each word of each review to a topic and thus identifies the topics present in each review, returning as output a probability distribution of the topics for each review and a list of the most probable words per topic.
For each dataset, it was necessary to create two different elements to be used in the construction of the model:

- **a dictionary**: a list of all the terms present within the dataset used;
- **a bag of words model**: called 'corpus' in the notebook, it is nothing more than a numerical representation of the reviews of the dataset in which each review is represented as a vector of term frequencies.

After several attempts, it was decided to use 8 as a parameter for the number of topics to be used.
To assess the goodness of the models, coherence was used: this is a measure of the quality of the topics generated by the model based on how the terms within a topic relate to each other. The higher the coherence, the greater the probability that the terms within a topic are semantically or semantically related to each other.

5.2 TF-IDF topic modelling

Dataset	Coherence
Negative (0)	0.432
Positive (1)	0.382

Table 3: *evaluation of topic modelling with the dataset obtained with the TF-IDF representation*

in addition to the coherence measure, the first three words that appear most frequently in the topics are reported:
Top-3 most salient terms in the model with negative classifications for each topic:

Topic	First word	Second word	Third word
1	Book	Read	One
2	One	Like	Movie
3	One	Game	Get
4	Movie	Dvd	Good
5	Product	One	Quality
6	Get	Would	Item
7	Would	One	Music
8	Product	Buy	Money

Table 4: *terms most present in the model with negative classifications*

The visualization of the three most recurrent words in each topic gives us the possibility to understand what the different topics were modelled on: it is evident that there are five distinct topics based on different products, namely **books** (topic 1), **movies** x2 (topics 2 and 4), **games** (topic 3) and **music** (topic 7). Two topics, on the other hand, seem to be much more generic, with a dense presence of the words 'item' (topic 6) and 'product' (topic 8): these are probably negative reviews concerning the quality of products without ever specifying which one they are. Moreover, it is very curious that the term 'one' is recurrently present in 5 of the 8 modelled topics.

Top-3 most salient terms in the model with positive classifications for each topic:

Topic	First word	Second word	Third word
1	Book	Read	Great
2	Great	One	Good
3	Good	Cd	One
4	Book	Great	Good
5	Game	Movie	Like
6	One	Book	Good
7	Great	Like	Time
8	Set	Great	Love

Table 5: *terms most present in the model with positive classifications*

The visualization of the 3 most recurrent terms in the topic modelling constructed with the dataset of reviews classified as positive also gives us the possibility to identify 5 topics modelled on the basis of products: **Books** x3 (topics 1, 4 and 6), **CDs** (topic 3), **movies** (topic 5): we can note that for the first time in one topic (number 5), two different product categories appear to be very recurrent. These are **Game** and **Movie**, and this event can be justified by two ideas: the first is that there is a comparison between Game inspired by Movie, and the second is that these two products are evaluated on the basis of the same qualities, which is why they are highly recurrent within the topic. It would be interesting to analyse this further to understand why.
Here again, we find a high presence in the topics of the term 'one' and we can also find a large presence of terms expressing a positive judgement such as 'great', 'good', 'like', 'love', which testify to the goodness of the positive classification.

5.3 Doc2Vec topic modelling

Dataset	Coherence
Negative (0)	0.408
Positive (1)	0.402

Table 6: *evaluation of topic modelling with the dataset obtained with the Doc2Vec representation*

As was done for the dataset built on the classifications obtained using the TF-IDF text representation, the analysis of the three most relevant terms in the different topics obtained using the dataset built on the classifications obtained on the text representation using Doc2Vec is conducted:

Top-3 most salient terms in the model with negative classifications for each topic:

Topic	First word	Second word	Third word
1	Would	Like	One
2	Book	Read	Like
3	Book	Read	One
4	Movie	Get	Product
5	One	Cd	Album
6	Movie	film	One
7	Book	Good	Read
8	Product	Well	One

Table 7: *terms most present in the model with negative classifications*

Again, it is possible to identify a certain trend in shaping topics on the basis of the product reviewed: **book** x3 (topic 2, 3, and 7), **movie** x2 (topic 4 and 6)and **CD** (topic 5). There are also three 'misleading' topics that present terms that can be associated with a liking of the product, such as like, well and good: it should be pointed out, however, that these could also be vitiated by errors created by the text classification model in question.

Top-3 most salient terms in the model with positive classifications for each topic:

Topic	First word	Second word	Third word
1	Cd	One	Album
2	Book	Read	Movie
3	Book	Read	Great
4	Great	Really	Get
5	Great	Good	Use
6	One	Great	Like
7	Game	Great	Good
8	Great	Book	Like

Table 8: *terms most present in the model with positive classifications*

Also confirmed in this fourth and final analysis of the three most recurrent terms in the generated topics is the trend towards topic modelling dependent on the product category: **CD** (topic 1), **book** (topics 2 and 3), **Game** (topic 7).
It is possible to note that we again find two different product categories among the most recurrent terms in the same topic: in topic 2, in fact, the terms **Book** and **Movie** are very recurrent. This can be justified by the idea that most of the reviews in the topic refer to purchases which compare books and films inspired by each other.
Also evident is the dense presence of expressions of positive judgements, which tend to witness how much reviews have been correctly classified previously.

6 Conclusions and future developments

This section of the document addresses a question I asked myself: is it possible to give a business value to the models used for this project?
The entire pipeline is based on categorizing reviews in order to create two sub-portions to understand which topics are most recurrent in positive and negative reviews. So why not think about using this to understand what aspects need improvement and what strengths to draw inspiration from in both review categories?
At present, the key to the success of any service like Amazon (which holds the data used), is the relationship with customers. Studying their habits and how they evaluate services gives the opportunity to make decisions that strengthen customer fidelity.

The idea, therefore, for a hypothetical future development could be to enrich the project by adding:

- A process that can identify which reviews were written by bots in order to remove them and make the dataset more truthful;
- Refining the performance of topic modeling models by also focusing on a subsequent semantic analysis of the reviews associated with each topic in order to make decisions that improve the service, such as quality control for some product categories, better packaging for safer transport, a faster or more efficient delivery service.

7 Libraries used

- pandas: `https://pandas.pydata.org/docs/`;
- numpy: `https://numpy.org/doc/`;
- seaborn: `https://seaborn.pydata.org/`;
- BZ2: `https://docs.python.org/3/library/bz2.html`;
- re: `https://docs.python.org/3/library/re.html`;
- os: `https://docs.python.org/3/library/os.html`;
- nltk: `https://www.nltk.org/`;
- gensim: `https://radimrehurek.com/gensim/auto_examples/index.html`;
- pyldavis: `https://pyldavis.readthedocs.io/en/latest/`;
- sickit-learn: `https://scikit-learn.org/stable/`;
- matplotlib: `https://matplotlib.org/stable/index.html`;
- keras: `https://keras.io/`;
- pickle: `https://docs.python.org/3/library/pickle.html`;
- scipy: `https://docs.scipy.org/doc/scipy/`;
- random: `https://docs.python.org/3/library/random.html`;

8 Bibliography

- Kamran Kowsari,Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes and Donald BrownText; Classification Algorithms: A Survey, 23 April 2019;
- Pooja Kherwa1, Poonam Bansal; Topic Modeling: A Comprehensive Review, 24 July 2019;
- Alper Kursat Uysal, Serkan Gunal; The impact of preprocessing on text classification, 16 September 2013;
- Mekhail Mustak, Joni Salminen, Loïc Plé, Jochen Wirtz; Artificial intelligence in marketing: Topic modeling, scientometric analysis, and research agenda, 2 November 2020;