Goodreads Review: Text Classification and Text Clustering Analysis

Text Mining & Search
Academic Year 2023/24

Marco Guarisco

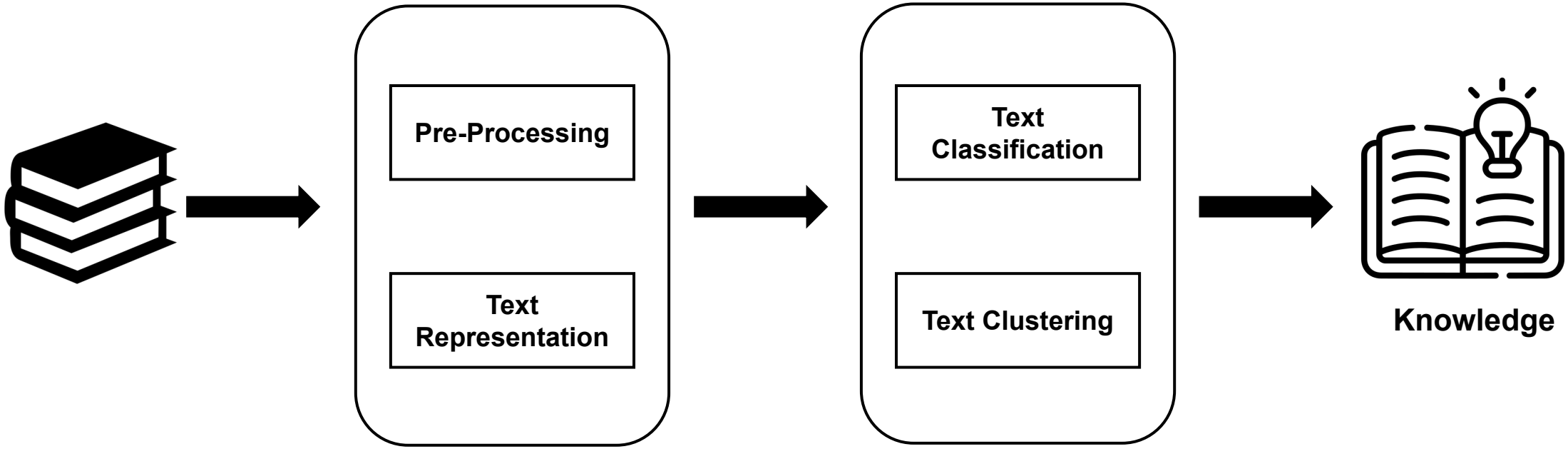
Marco Sallustio

Salvatore Rastelli

INTRODUCTION & TASK

The aim of the project is to analyze the *Goodreads* reviews through two Text Mining techniques: *Text Classification* and *Text Clustering*.

This is the workflow of our project:



DATASET

- **Goodreads dataset** is a big dataset of books
- This dataset is divided into three group including users' detailed book reviews
- These reviews are divided by genre
- We used those relating to *fantasy/paranormal* books
- Due to too high computational times we decided to work on a sampled version of the original dataset, consisting in 100,000 reviews.

PRE-PROCESSING (1)

In this phase we performed the following operations:

1. **Language Based Filtering** → English Language

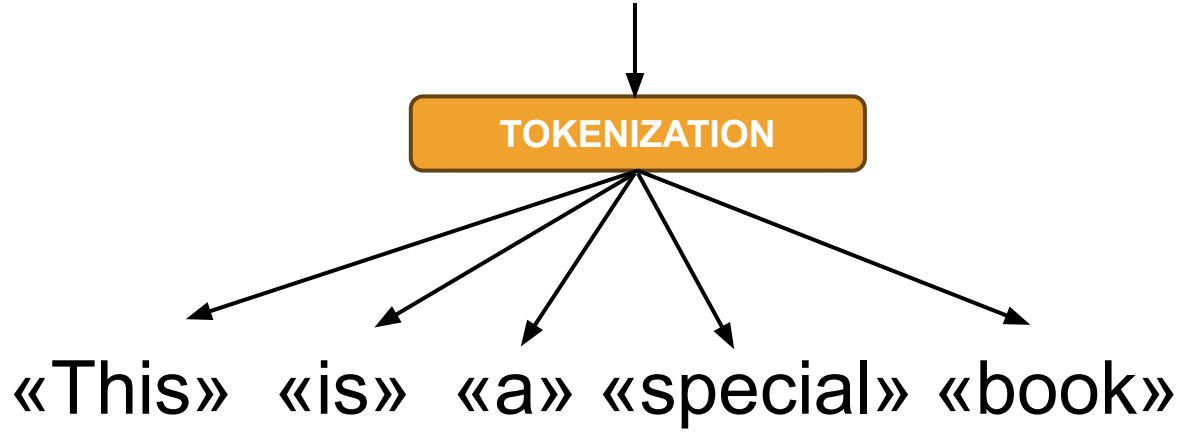
2. **Normalizing case** → Lowercase

3. **Remove numbers,punctuation and extra white spaces**;

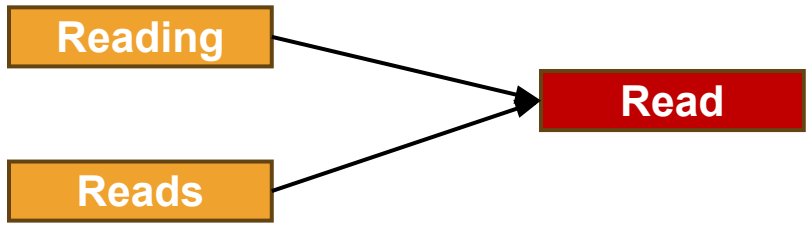
4. **Stopwords removal** → NLTK package

PRE-PROCESSING (2)

5. **Tokenization***:* «This is a special book»



6. **Lemmatization***:*



EXPLORATORY ANALYSIS

In this phase we did some exploratory analyses conducted on the dataset both before and after the Pre-Processing phase.

On the dataset before Pre-Processing phase are listed below the analysis executed:

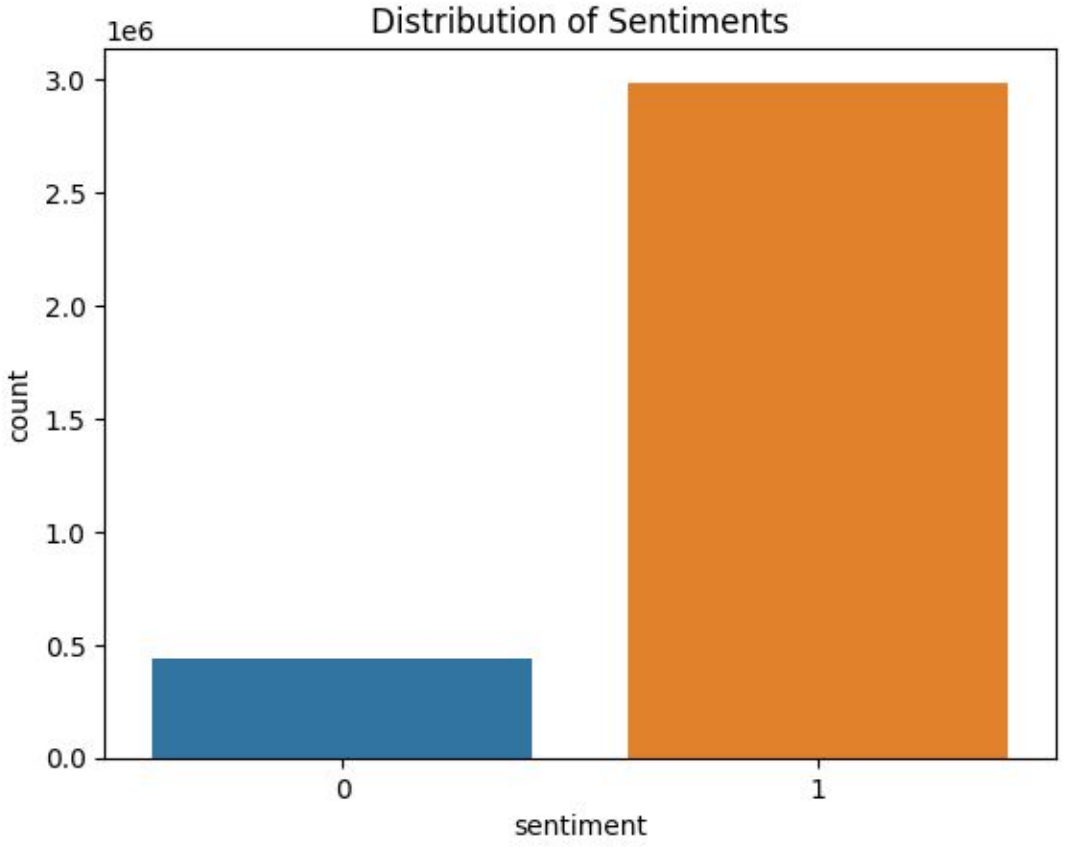
- **Distribution of Sentiment**→ to check the balance of the two classes: *NEG* and *POS*;
- **Distribution of Rating**→ to understand which rating was used the most;

On the dataset after Pre-Processing phase we added two additional analysis:

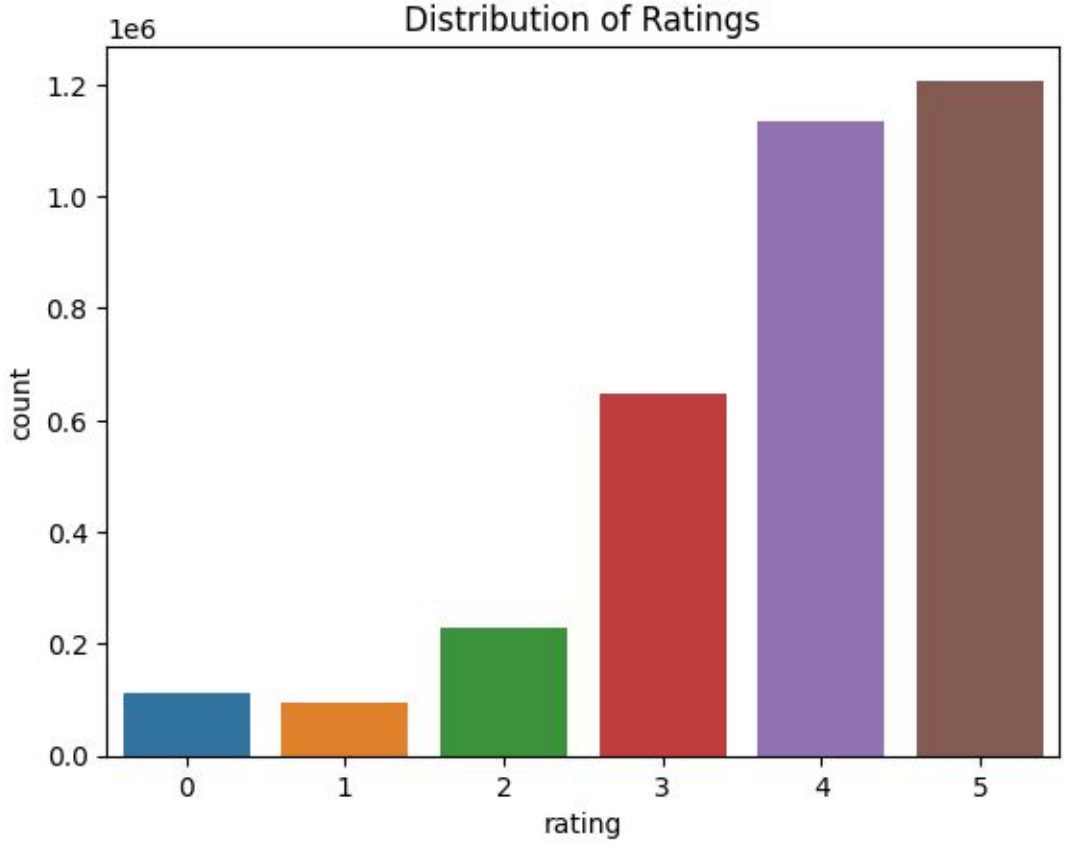
- **Average Length of Reviews by Rating**→ to understand if there are differences in the average length of reviews depending on the rating.
- **Tri-gram Analysis**→ to see if there are significant word associations;

EXPLORATORY ANALYSIS (PRE)

Distribution of Sentiment:

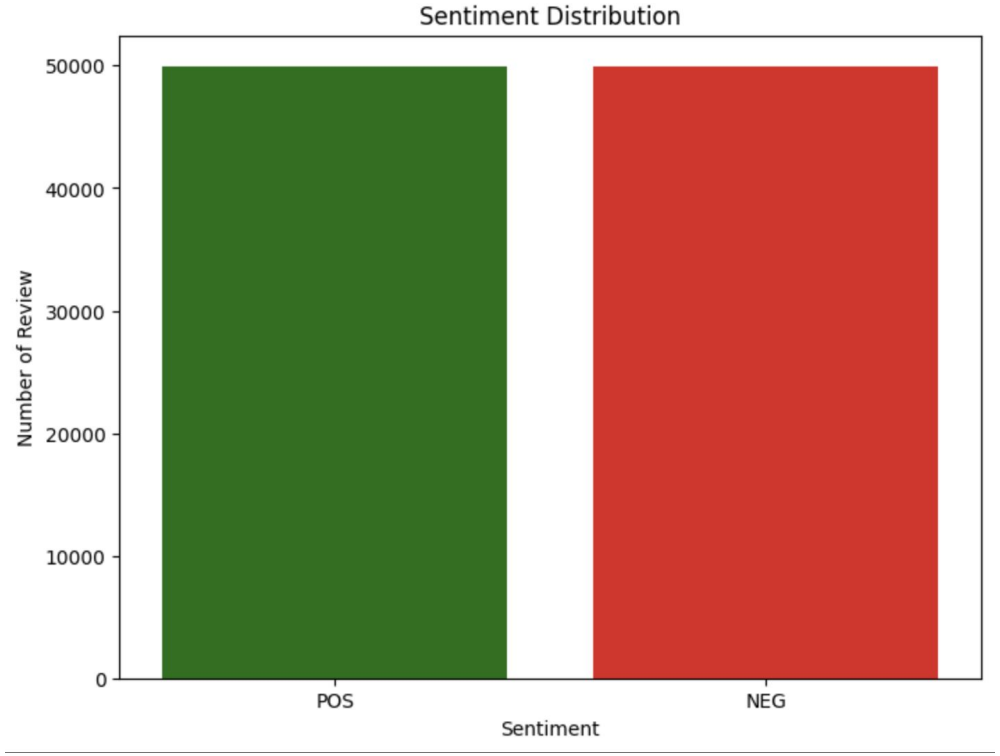


Distribution of Rating:

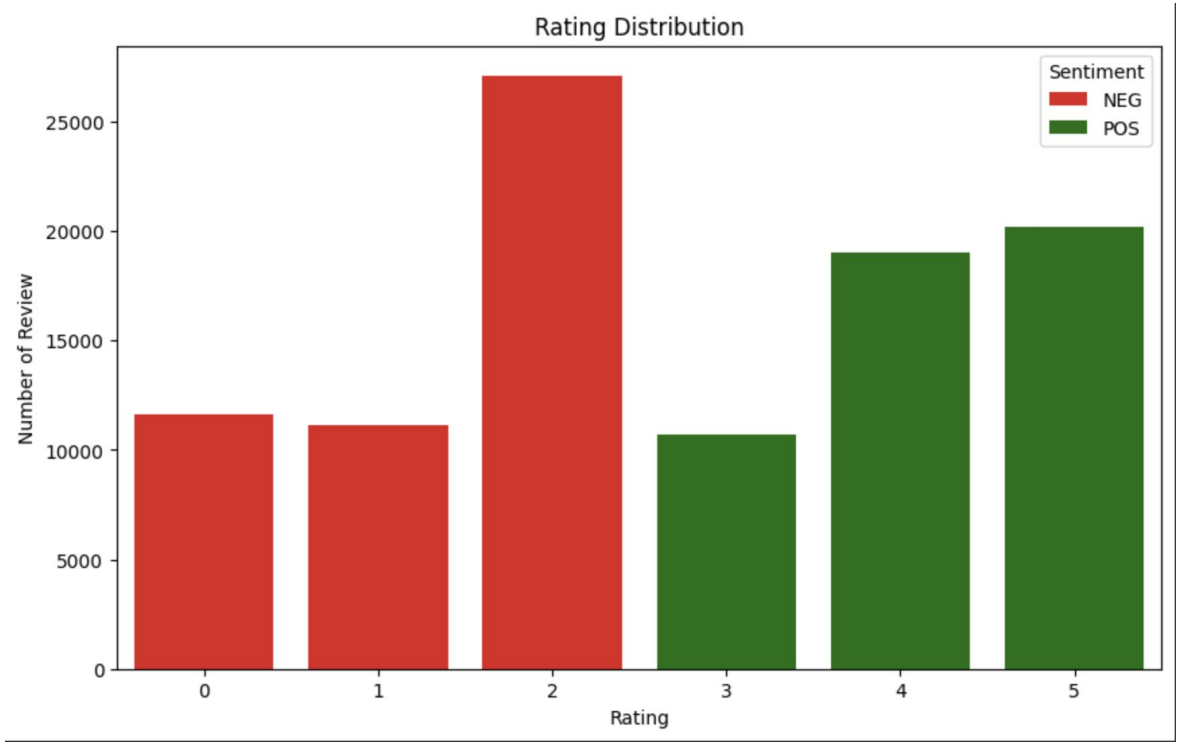


EXPLORATORY ANALYSIS (POST)

Distribution of Sentiment:



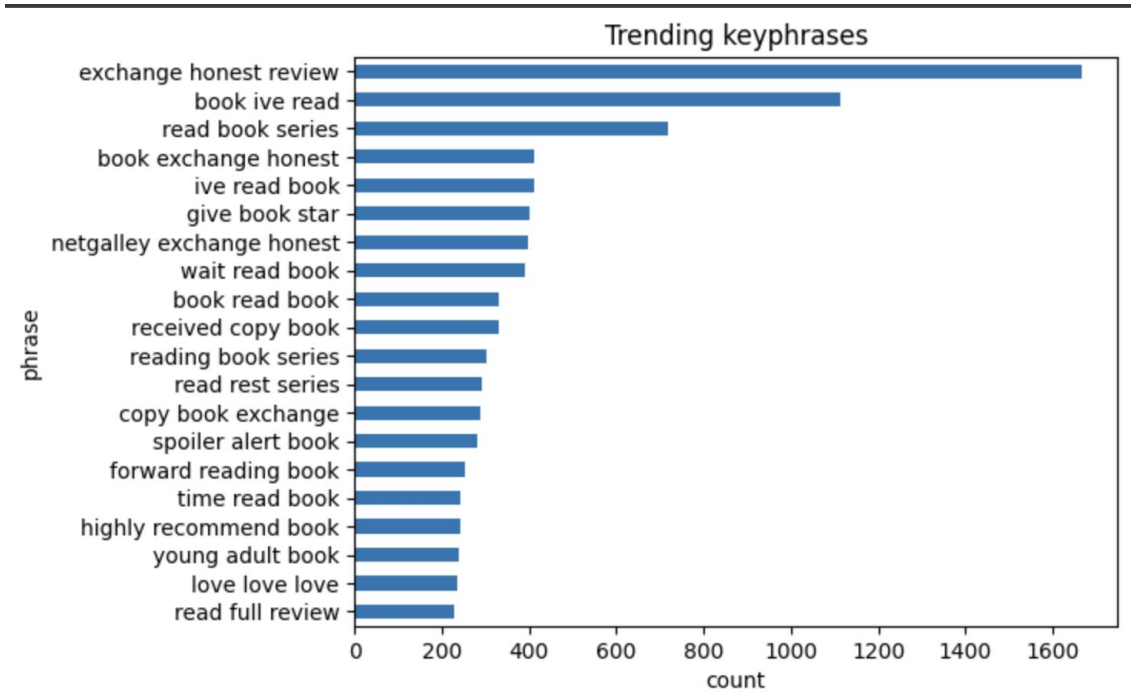
Distribution of Rating:



EXPLORATORY ANALYSIS (POST)

Average Length of Reviews by Rating :

Tri-gram Analysis:



TEXT REPRESENTATION

In order to train the classifiers, we need to get a numerical representation of the text:

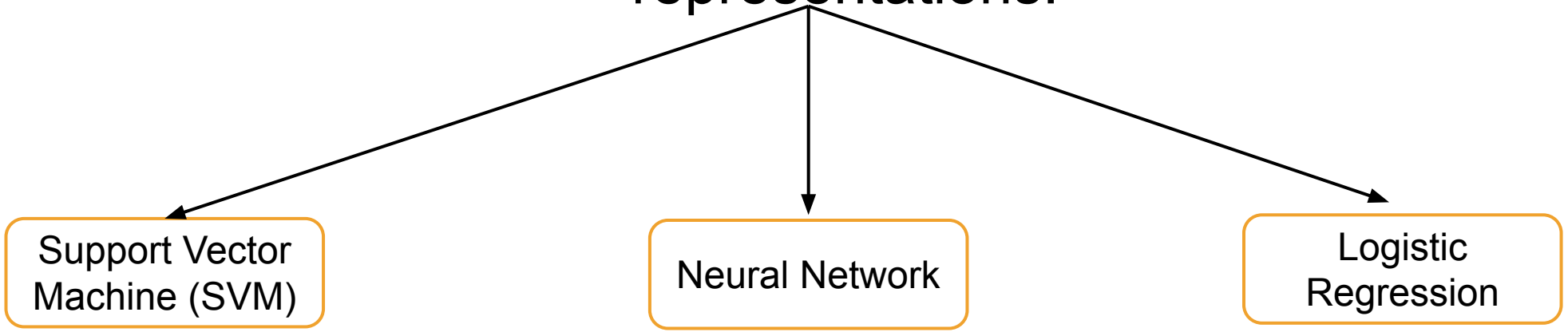
- **TF-IDF**: we considered uni-grams, bi-grams, and tri-grams, and the final vectorized text is composed by 1000 features.
- **Doc2Vec**: we first used the *TaggedDocument()* method which takes care of associating a unique tag to each review, and then *doc2vec()* method in which:
  - we passed the tagged reviews on the basis of which the model will be trained;
  - the size of the vector that must represent each review→1000;
  - 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 →1;
  - the number of threads to be used during the training process.

TASK 1: TEXT CLASSIFICATION

- In order to have a binary classification we created a new column with two possible values:

TASK 1: TEXT CLASSIFICATION

- The dataset was divided into Train(70%) and Test(30%).
- We used 3 different classification models on the 2 different text representations:



RESULTS

We have obtained the following results distinguishing the two different Text Representations:

TF-IDF

SVM

	precision	recall	f1-score	support
Positive	0.764	0.751	0.757	15101
Negative	0.751	0.765	0.758	14864
accuracy			0.758	29965
macro avg	0.758	0.758	0.758	29965
weighted avg	0.758	0.758	0.758	29965

MLP

	precision	recall	f1-score	support
Positive	0.757	0.707	0.731	15101
Negative	0.721	0.769	0.744	14864
accuracy			0.738	29965
macro avg	0.739	0.738	0.737	29965
weighted avg	0.739	0.738	0.737	29965

Logistic Regression

	precision	recall	f1-score	support
Positive	0.764	0.753	0.758	15101
Negative	0.753	0.764	0.758	14864
accuracy			0.758	29965
macro avg	0.758	0.758	0.758	29965
weighted avg	0.758	0.758	0.758	29965

RESULTS

Doc2Vec

SVM

	precision	recall	f1-score	support
Positive	0.752	0.731	0.741	15101
Negative	0.734	0.755	0.744	14864
accuracy			0.743	29965
macro avg	0.743	0.743	0.743	29965
weighted avg	0.743	0.743	0.743	29965

MLP

	precision	recall	f1-score	support
Positive	0.764	0.741	0.753	15101
Negative	0.745	0.768	0.756	14864
accuracy			0.754	29965
macro avg	0.755	0.755	0.754	29965
weighted avg	0.755	0.754	0.754	29965

Logistic Regression

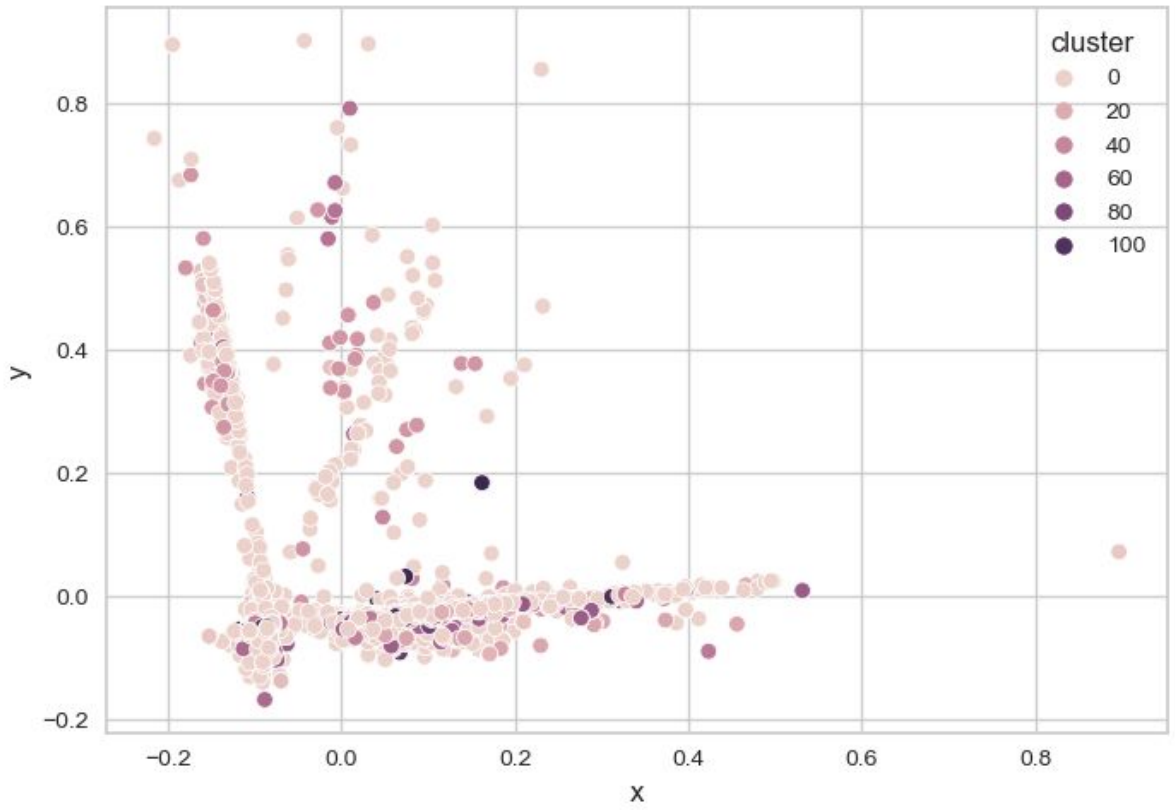
	precision	recall	f1-score	support
Positive	0.747	0.742	0.745	15101
Negative	0.740	0.745	0.742	14864
accuracy			0.744	29965
macro avg	0.743	0.744	0.743	29965
weighted avg	0.744	0.744	0.744	29965

TASK 2: TEXT CLUSTERING

For this phase we used two different algorithms:

- **K-Means**: a partitional, total and exclusive clustering algorithm. It assigns points to clusters based on a specific distance. It is needed to specify the number of clusters.
- **DBSCAN**: a partitional, partial and exclusive clustering algorithm. It creates clusters selecting dense regions of points. It is not needed to specify the number of clusters;

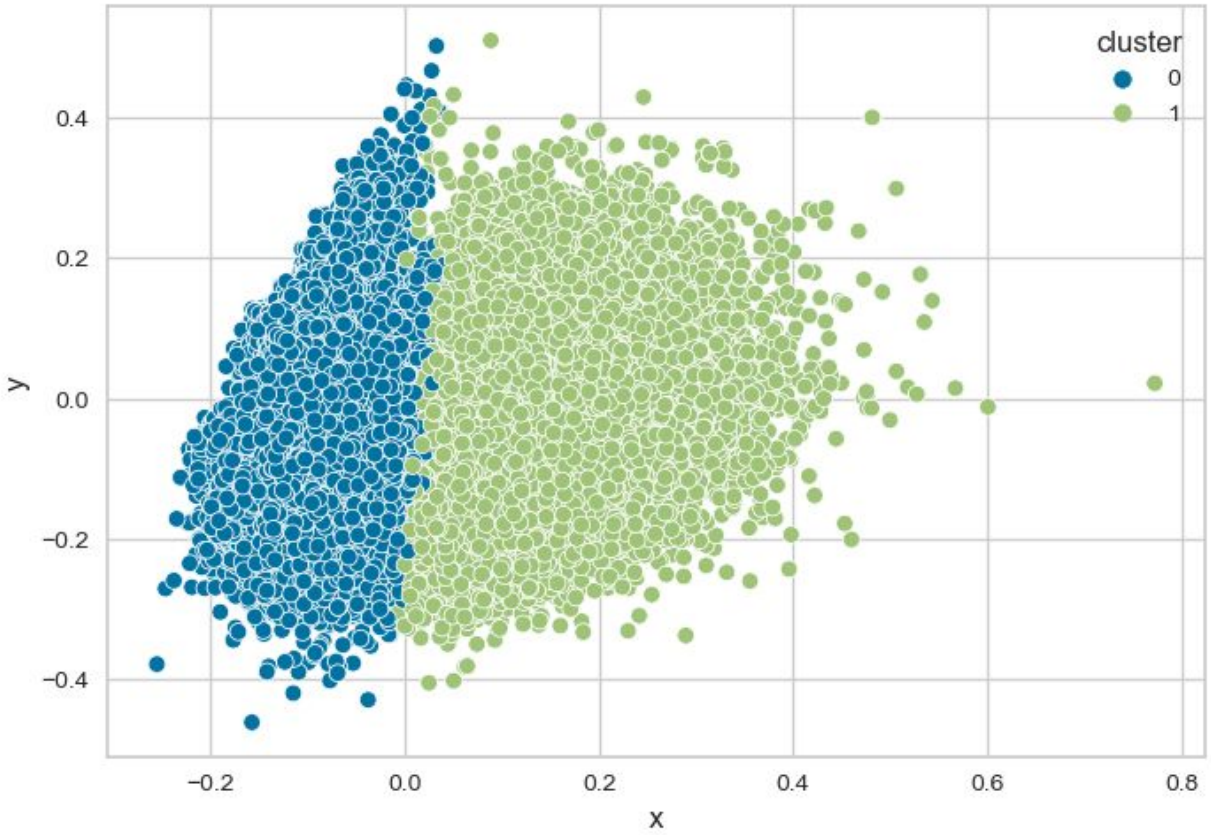
RESULTS DBSCAN



We set the minimum distance between two points that are in the same cluster equal to 0.4 and the minimum number of elements per cluster equal to 10.

The algorithm found a number of 118 clusters and 8832 observations were assigned to a cluster

RESULTS K-MEANS



We used the *Elbow method* to choose the value of K to initialize the algorithm: the chosen value was 2.

EVALUATION

We used the **Silhouette Coefficient** to evaluate the Algorithms and we obtained the following result:

	DBSCAN	**K-Means**
Silhouette Coefficient	0.204	0.020

The value for the **DBSCAN** algorithm is higher than for the **K-Means** algorithm, even though only 8800 reviews out of 10000 are assigned to a cluster. However, in both cases the value is near 0, suggesting that the clusters obtained are not well separated.

CONCLUSIONS

- We can say that in *Text Classification*, various models show acceptable binary class predictions, with the best performance observed using the Tf-Idf representation.
- *Text Clustering* on the dataset didn't substantially improve data understanding; both methods didn’t succeed in identify clusters well separated.