1. Introduction

Text mining is an artificial intelligence technology that uses natural language processing to transform the unstructured text in documents and databases into normalized, structured data suitable for analysis or to drive machine learning algorithms.

Natural language processing is an area of computer science and artificial intelligence that is concerned with the interaction between computers and humans in natural language. The ultimate goal of Natural language processing is to enable computers to understand language as well as we do.

This project can be seen as a text classification problem. Text Classification is one of the widely used NLP applications in different business problems

Text classifiers can be used to organize, structure, and categorize any kind of text – from documents, medical studies and file from all over the web. For example, new articles can be organized by topics; support tickets can be organized by urgency; chat conversations can be organized by language; brand mentions can be organized by sentiment; and so on.

Text classification is one of the fundamental tasks in natural language processing with broad applications such as sentiment analysis, topic labeling, spam detection, and intent detection

This project is intended to be a walkthrough of a machine learning project classification model that is able to predict the author of a specific book/article.

In this report, we are going to present the different methodologies such as data collection, cleaning, and preprocessing techniques, used to find out which is the best model for predictions to find who was the Portuguese author who wrote these transcripts.

2. Corpora Description

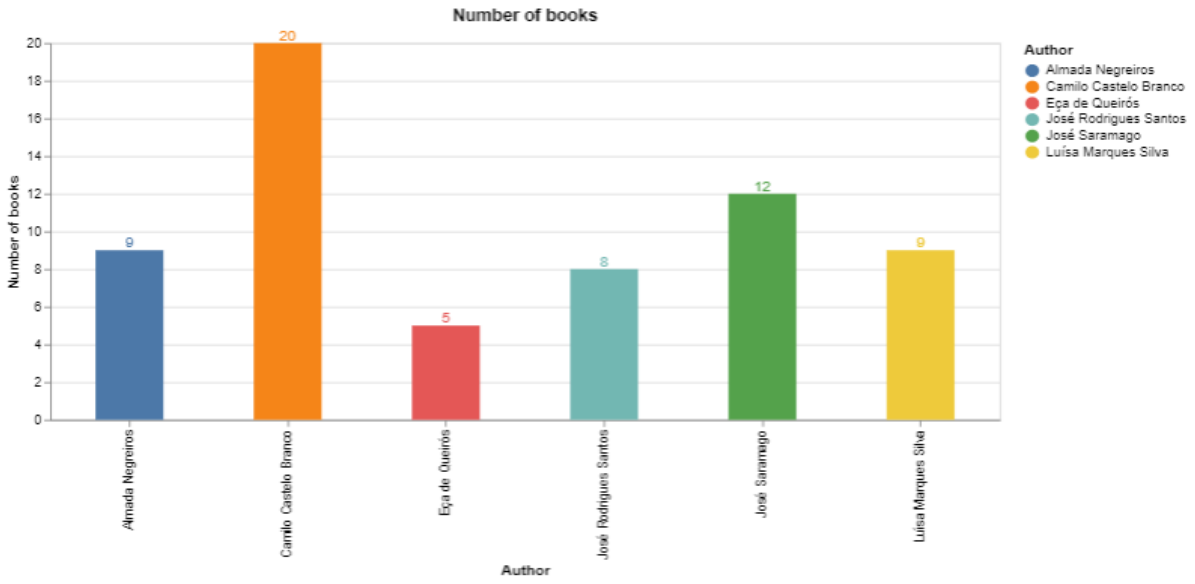
A corpora is a collection of written texts, especially the entire works of a particular author or a body of writing on a particular subject. In this report, we have transcripts from many Portuguese authors.

It is a common practice to carry out an exploratory data analysis in order to gain some insights from the data. However, up to this point, we don't have any features that define our data.

One of our main concerns when developing a classification model is whether the different classes are balanced. This means that the dataset contains an approximately equal portion of each class. For example, if we had two classes and a 95% of observations belonging to one the classes, a classifier will always output the majority class % failling all the predictions of the minority class.

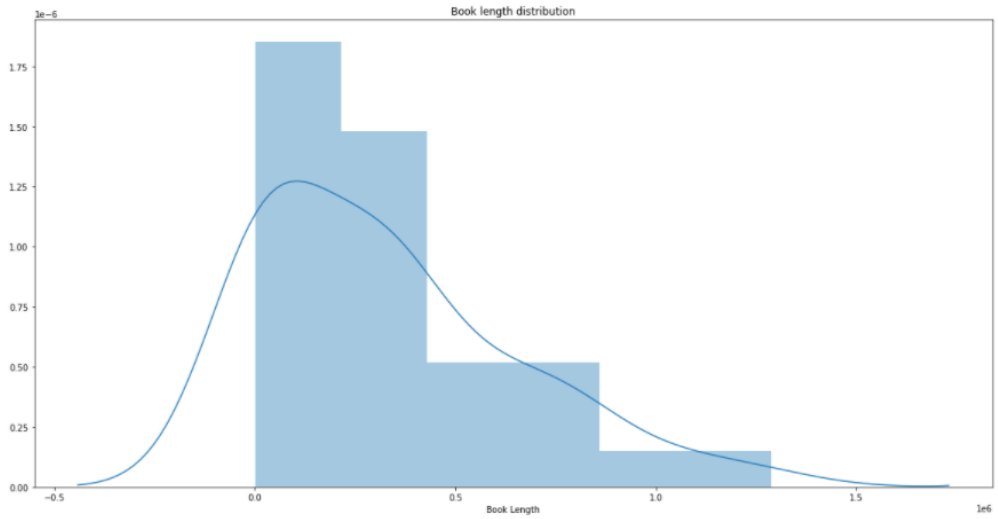
There are several ways of dealing with imbalanced datasets. One first approach is to undersample the majority class and oversample the minority one, so as to obtain a more balanced dataset. Other approach can be using other error metrics beyond accuracy such as the precision, the recall or the F1-score. We will talk more about these metrics later.

Looking at our data, we can get the number of observations belonging to each class:



From this simple count plot, we can see that our corpora consist of 63 transcripts from 6 different authors.

Another variable of interest can be the length of each of the transcripts. By plotting the distribution plots, it shows signs of a distribution that is positive skewed.



3. Pipeline Description

3.1 Types of text pre-processing techniques

When dealing with numerical data, data cleaning often involves removing null values and duplicate data, dealing with outliers, etc. With text data, there are some common data cleaning techniques, which are also known as text pre-processing techniques.

Noise removal is about removing characters, digits and pieces of text that can interfere with your text analysis. It is one of the most essential text preprocessing steps and one of the first things you should be looking into when it comes to Text Mining.

There are various ways to remove noise. This includes punctuation removal, special character removal, numbers removal, html formatting removal, domain specific keyword removal and many more.

Before creating any feature from the raw text, we must perform a cleaning process to ensure no distortions are introduced to the model. We have followed these steps:

- **Special character cleaning:** special characters such as “\n” must be removed from the text since we are not expecting any predicting power from them.
- **Lower casing characters:** we would expect, for example, “Livro” and “livro” to be the same word and have the same predicting power. For that reason, we have lower cased every word.
- **Punctuation signs**: characters such as “?”, “!”, “;” have been removed.

- **Stemming or Lemmatization**: stemming is the process of reducing derived words to their root. Lemmatization is the process of reducing a word to its lemma. The main difference between both methods is that lemmatization provides existing words, whereas stemming provides the root, which may not be an existing word. So, for our analysis we only considered lemmatization.
- **Stop words:** words such as "de" or "a" won't have any predicting power since they will presumably be common to all the documents. For this reason, they may represent noise that can be eliminated. We have downloaded a list of portuguese stop words from the nltk package and then deleted them from the corpus.

With each cleaning process applied we created a new column in our data frame that reflects each one of these cleaning steps.

4.2 Author Coding

Machine learning models require numeric features and labels to provide a prediction. For this reason, we must create a dictionary to map each label to a numerical ID. We have created this mapping scheme:

Author	Category code
Luísa Marques Silva	1
José Saramago	2
José Rodrigues Santos	3
Eça de Queirós	4
Camilo Castelo Branco	5
Alamada Negreiros	6

4.3 Text Representation

To represent our text, every row of the dataset will be a single document of the corpus. This will vary depending on the method that we choose:

- **Word Count Vectors:** With this method, every column is a term from the corpus, and every cell represents the frequency count of each term in each document.
- **TF–IDF Vectors:** score that represents the relative importance of a term in the document and the entire corpus. The TF-IDF value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain

the word, which helps to adjust for the fact that some words appear more frequently in general.

These two methods (Word Count Vectors and TF-IDF Vectors) are often named Bag of Words methods since the order of the words in a sentence is ignored. Other text representation techniques include:

- **Word Embeddings:** The position of a word within the vector space is learned from text and is based on the words that surround the word when it is used. Word embeddings can be used with pre-trained models applying transfer learning.
- **Topic Modes:** Methods such as Latent Dirichlet Allocation try to represent every topic by a probabilistic distribution over words, in what is known as topic modeling

We have chosen TF-IDF vectors to represent the documents in our corpus. This election is motivated by the following points:

- TF-IDF is a simple model that yields great results in this area.
- TF-IDF features creation is a fast process, which will lead us to shorter waiting.
- We can tune the feature creation process in python to find the best results.

4.4. Train – test split

We need to set apart a test set to prove the quality of our models when predicting unseen data. We have chosen a random split with 70% of the observations composing the training test and 30% of the observations composing the test set.

5. Predictive Models

We have tested several machine learning models to figure out which one may fit better to the data and properly capture the relationships across the points and their labels.

We have tried the following models:

- Random Forest

- Support Vector Machine
- K Nearest Neighbors
- Multinomial Naïve Bayes
- Logistic Regression

5.1 Evaluation and Results

In this final phase we are going to evaluate the results our built models using the evaluation metrics available at the scikit/learn package. We considered the following metrics:

Accuracy Score: Accuracy score is one of the most basic evaluation metrics which is widely used to evaluate classification models. The accuracy score is calculated simply by dividing the number of correct predictions made by the model by the total number of predictions made by the model. The formula is presented below:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision: Precision measures how accurately the model can capture fraud i.e., out of the total predicted fraud cases, how many turned out to be fraud.

$$Precision = \frac{TP}{TP + FP}$$

Recall: Recall measures out of all the actual fraud cases; how many the model could predict correctly as fraud. This is an important metric here.

$$Recall = \frac{TP}{TP + FN}$$

F1-score: this is a balance between precision and recall.

$$F1 - Score = 2 * \frac{(precision * recall)}{precision + recall}$$

In this application, we just want documents to be correctly predicted. For this reason, it does not matter to us whether our classifier is more specific or more sensitive, as long as it classifies correctly as much documents as possible. Therefore, we have studied the accuracy when comparing model calculating both the accuracy on both training and test. However, we have also obtained the confusion

matrix and the classification report (which computes precision, recall and F1-score for all the classes) for every model, so we could further interpret their behavior.

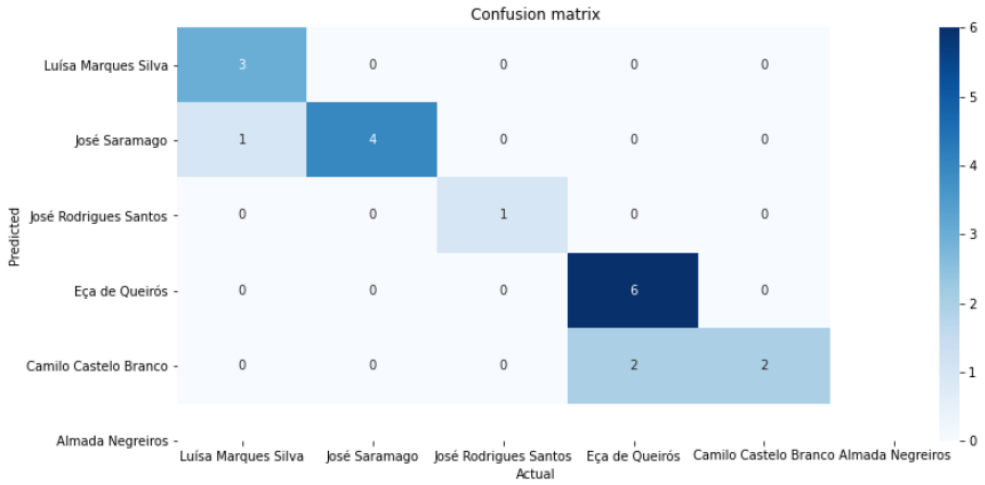
5.2 Best Model Selection

Below we show a summary of the different models and their evaluation metrics:

Model	Training Set Accuracy	Testing Set Accuracy
Random Forest	100%	84,21%
Support Vector Machine	100%	57,89%
K Nearest Neighbors	93,18%	73,68%
Multinomial Naïve Bayes	68,18%	52,63%
Logistic Regression	95,45%	57,89%

Overall, we obtain good accuracy values for the Random forest and approximately the KNN model. We can observe that the remaining models performed poorly in predicting the correct author of the text and they also seem too overfit since they have an extremely high training set accuracy but a lower test set accuracy.

We will choose the Random Forest classifier above the remaining models because it has the highest test set accuracy, which is near to the training set accuracy. The confusion matrix and the classification report of the Random Forest model are the following:



Classification report

	precision	recall	f1-score	support
1	0.67	0.67	0.67	3
2	1.00	0.80	0.89	5
3	1.00	1.00	1.00	1
5	0.75	1.00	0.86	6
6	0.67	0.50	0.57	4
accuracy			0.79	19
macro avg	0.82	0.79	0.80	19
weighted avg	0.80	0.79	0.78	19

Analyzing the confusion matrix of the Random Forest there were only three instances where the model did not predict correctly the author associated with the transcript.

Conclusions

A large collection of documents may provide useful information to anyone. But it is also a challenge to find out the useful information from a large collection of documents. Successfully implemented text mining techniques help to identify the category of each document.

The purpose of this report was to suggest an approach that can be used to identify the authors from text documents written in the portuguese language based on their content.

This report has done several interesting experiments based on text categorization. The data pre-processing stage prepared the data by removing any type of characters that may affect the model's performance. We then applied different machine learning techniques to obtain properly optimized results.

Text mining is helping companies become more productive by using these insights to make data-driven decisions.

Many time-consuming and repetitive tasks can now be replaced by algorithms that learn from examples to achieve faster and highly accurate results. The possibility of analysing large sets of data and using different techniques, such as sentiment analysis, topic labelling or keyword detection, leads to enlightening observations about what customers think and feel about a product.