Genre Identification on Gutenberg Corpus

Joel John Philip
Fakultät Für Informatik
Otto von Guericke University Magdeburg, Germany

Geetha Doddapaneni Gopinath
Fakultät Für Informatik
Otto von Guericke University Magdeburg, Germany

Rekha Nuggehalli Sundrakrishna
Fakultät Für Informatik
Otto von Guericke University Magdeburg, Germany

Suhas Shantharam
Fakultät Für Informatik
Otto von Guericke University Magdeburg, Germany

***Abstract*—This paper is the documentation of an attempt to build a model that is capable of classifying the books into different genres. Classification of textual data is challenging as we need to consider the semantics and the syntax with equal importance. Based on this idea the features have been extracted and suitable model has been trained . On a whole, 16 different features have been extracted which is discussed in the concept section, followed by the implementation of these features to train a model and the evaluation and results represented. A final conclusion on which model performed the best has been discussed at the end.**

***Index Terms*—text classification model, Project Gutenberg, feature extraction, genre identification**

I. Introduction and Motivation

Book reading is known to be one of the oldest hobbies and for people who love and enjoy reading experience a journey with the book. That is when it becomes important for the reader to find the perfect book that would bring out the pleasurable experience to them. It is well known that finding the right book takes a lot of effort, in fact finding a book in general is a hassle. This is the main motivation behind our project, where we have tried to build a model which makes it easy for the user to search for books of their desired type. One of the most common search methods used to find a book is using genres. And in order to get search results for a particular genre we need accurate classification of books.

In this project , we first explore the dataset and extract relevant features that are further used to learn an efficient text classification model for genre identification. In the further sections we would be discussing in detail the general structure of the data, its features and possible issues. Also we would be doing a comparison among different classification models.

II. Dataset

According to the problem statement The Project Gutenberg has been used as a text corpus to train classifiers, with a goal of being able to identify the genre of a given book.Among the major linguistic datasets The Gutenberg Corpus is one of the most popularly used dataset for text classification tasks because of its vast collection of e-books and also due to the free access of the dataset on the internet. Digging further into the dataset, it is a collection of over 60,000 books authored by 142 different authors. For this project we have used books belonging to the 19th Century English Fiction which is a part of Project Gutenberg. It is a small collection of around 1,000 books stored as html files of 10 various genres.

A little deeper into the characteristics of the dataset, it consists of a separate file containing details such as book name, book ID, genre and the author name. The 10 genres being ’Allegories’, ’Christmas Stories’, ’Detective and Mystery’, ’Ghost and Horror’, ’Humorous and Wit and Satire’, ’Literary’, ’Love and Romance’, ’Sea and Adventure’, ’Western Stories’ are to be identified by the classifier. Along with analysing the characteristics of the documents in the dataset, it is important to gain an understanding of the contents of the books and how they differ across genre.

The dataset has instances which are unevenly distributed i.e one of the classes has higher number of instances as compared to other classes as shown in the figure. Training a model with such a dataset will affect the learning of the classifier. Hence handling this was one of the most important task in this project.

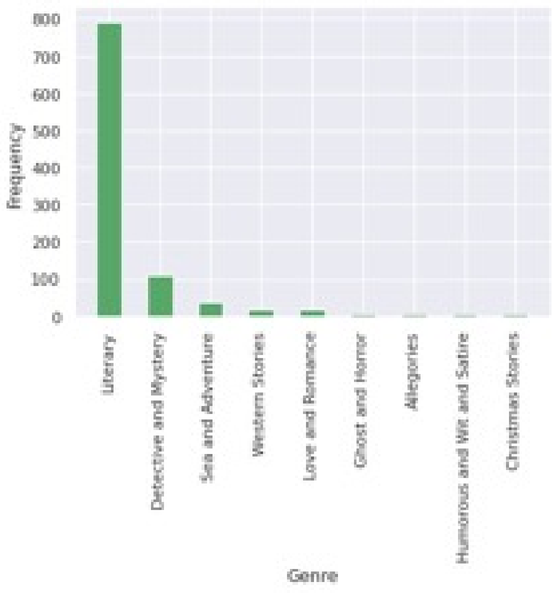

Fig. 1. Genre distribution in 19th century English fiction

III. Concept

Overall Concept: One of the main step in being able to train a good model that understands underlying properties of a document is to extract appropriate features. The dataset allows us to extract ample number of useful features such as writing style, sentence complexity, gender identification, setting, sentiment, plot complexity, lexical richness, ease of readability and so on that are useful for a classification problem. These features can be further simplified and divided into Readability, Paragraph count, male, female, type token ratio, names, pronouns, tenses (past, present, future), rural or urban setting and others. Below we discuss about some of the features that we have extracted to train our classifier for genre identification.Features being represented here are scaled on the basis of number of chunks a book gets divided into.

- Sentiment analysis: Identifying the polarity of the book is a basic task to classify a certain book as positive, negative or neutral. Sentiment tells the mood of the plot and at times changes during the course of the entire plot. For this reason it is necessary to do sentiment analysis for chunks of the data rather than on the entire text.
- Ease of readability: Emphasising on the writing style of the author, ease of readability is one of the major qualities to identify a certain book belonging to a certain writer. In its broadest sense a number of factors such as choice of words, voice, cultural reference, number of characters, inclusion of words of a different language and content, add to the complexity of a text. Flesh readability ease score has been employed for measurement of the same.
- Paragraph count: It is surprising to know that paragraph count of a text also plays a vital role in identifying the style of writing. Hence count of the number of paragraphs has been extracted for each text document.
- Part of speech (POS) tagging: POS tagging also called grammatical tagging is a process of marking a word in the text corpus to a particular part of speech based on context and definition. Although seems simple, identifying parts of speech could be exhausting for even a word as simple as a “pen” , which can be mistaken for a noun, if not provided with an appropriate context. For example, “I would like a blue pen” and “I would like to pen a letter” have the same word “pen” acting as a noun in the first sentence and as a verb in the second one. This proves that context acts as an important aspect in POS tagging. In our code we have identified words into four groups : nouns, verbs, adjectives and adverbs.
- Character count: The number of characters in a story adds to the complexity of the novel and also plays a major role in identifying the writing style of the author. Some authors use a lot of characters in the plot in contrast to some other authors preferring to include minimal characters. Important aspect is that a character can be addressed in many different variants such as first name, last name, full name or a nick name.
- Gender identification: Each author tends to add a certain gender orientation to the plot of the book, i.e by having more number of male characters or higher ratio of female characters. The story could be male or female oriented and this could be considered as one of the styles of the author, making it one of the necessary feature, therefore, gender identification is done on the list of characters extracted by character count and is ensured to not be case sensitive.
- Writing style: One of the many ways to analyse the style of writing is by the choice of words and their usage such as tenses, pronouns, conjunctions alongside some other features as referred before. We have chosen future, past and present tense, possessive and personal pronouns and coordinating conjunctions and extracted them on the created chunks of data.
- Lemmatisation: Process of grouping words and their different forms into a single category indicating various usages of the same words in different form, context or tense. It is done on the chunks of data obtained though it has the same effect when done on the whole text.
- Type Token Ratio: This gives us the number of unique words in the text and closer it is to one, the higher the lexical richness of the text. It is done on the list obtained after lemmatisation of chunks. This as mentioned before also adds to the writing style of the author.
- Setting: It is the time and location of the story and initiates the main backdrop and mood of the story. The combination of place, historic time and the environment provides general background for the characters of the plot. We should note that the setting may change depending on the events or scenes of the story. Hence have used the names of the characters and the quotes assigned to each character.

IV. Implementation

The implementation of the project is done in Python environment by making use of several libraries and frameworks which provided a broad variety of linguistic pre-processing, feature selection and standard machine learning models to apply the selected features on the main classification task. Below is the list of libraries and/or frameworks used in our project:

- NLTK
- Scikit Learn
- Pandas
- Numpy
- Matplotlib
- Seaborn
- Gender Guesser
- imblearn
- Collections

A. *User defined functions*

1) *chunking* - This function has been used to divide the entire text into a set of 10,000 tokens. Token list is passed as the input.

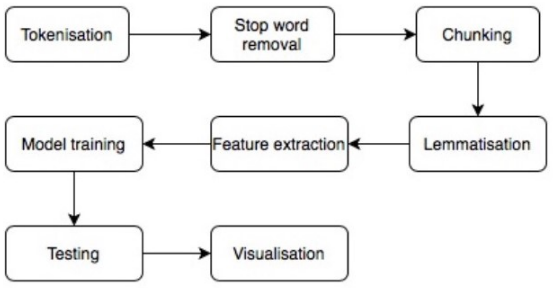

Fig. 2. Implementation pipeline

2) *sentiment analysis* - using sentiment intensity analyser, this function assigns positive, negative and neutral score to the chunk list passed as input.
3) *FleschReadabilityEase* - Takes the text as input and generates the readability ease value using the formula below :
   RE = 206.835 – (1.015 x ASL) – (84.6 x ASW)
   where,
   RE = Readability Ease
   ASL = Average Sentence Length (i.e., the number of words divided by the number of sentences)
   ASW = Average number of syllables per word (i.e., the number of syllables divided by the number of words)
4) *paragraph_count* - used to count the number of paragraphs by taking the text document as input.
5) *get_wordnet_pos* - used to identifying part of speech of the tokens in the text. It takes words as input.
6) *lemmatise* - used to get root words
7) *stopwordremove* - using an in-built English stop word dictionary of NLTK to remove all the stop words and character names.
8) *Charactertagger* - we used StandfordNERTagger inbuilt function of the NLTK library to identify the characters introduced in the book.
9) *TTR* - gives normalised value of total unique words present in the chunks.
10) *genderidentify* - takes characters as input and identifies the gender (male/female) of each character present in the book using a function detector of gender guesser library.
11) *writing_style* - it helps to determine the writing style of the author by identifying the tenses, pronouns and conjunctions used.
12) *feature_extractor* - is a function which calls all the above functions to extract the features.
13) *process_text* - it is for processing all text documents and making a csv out of extracted features of all files.
14) *cm_analysis* - is used to generate confusion matrices for the predictions made by the model.
15) *EDAVisualisation* - is for plotting exploratory graphs from the data.
16) *gridsearch* - Grid-search is used to find the optimal hyper-parameters of random forest model which results in the most 'accurate' predictions.
17) *preprocess* - since there is a lot of imbalance in the classes, this function uses SMOTE to generate synthetic data for the minority classes.
18) *evaluate* - for evaluation for the model predictions.
19) *train_model* - trains the Random Forest Classifier model.
20) *predict_genre* - is to make predictions using the trained model.

We experimented with various models but Random Forest Classifier gave the maximum scores such as accuracy, precision, recall and f1 score.

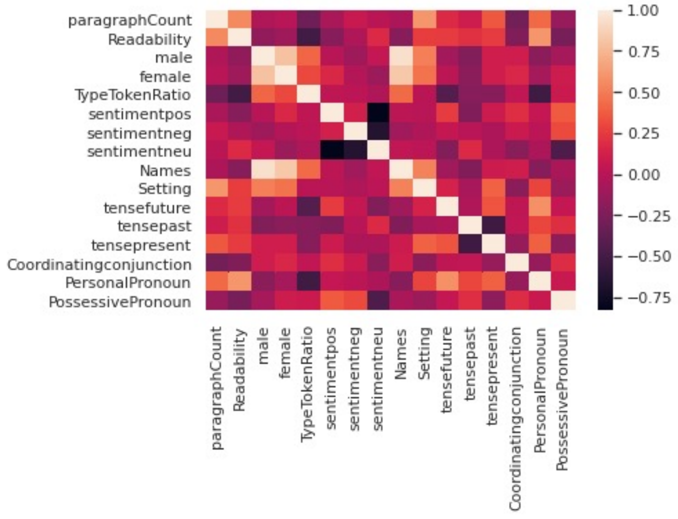

Fig. 3. Heatmap showing extracted feature distribution

V. Evaluation

A confusion matrix has been generated to represent the predictions and in figure 5 we can see that due to the imbalance in the class distribution, the classifier was overfitting and the rate of predictions into one of the classes is high. But in figure 6 the imbalance has been handled and the model does not overfit, hence better accuracy and precision rates during prediction and also better distribution of the predictions among the classes.On test data with a split of 0.2, the following metrics were achieved:Accuracy:83.550%, F1Score:0.8125555523920325, Precision Score:0.838796309, Recall Score:0.80252207610, Balanced accuracy score:0.8025220761.

Conclusion

The model with the feature extractor approach does indeed provide a better sense of accuracy and understanding of the top-k results retrieved when an instance of a book is queried compared to a "Bag of Words" approach wherein only the matching words are taken into consideration without the context or the semantics of the data at hand . It can also be seen that out of all the classifiers used, Random Forest has given the best prediction results. However there is more scope on offer in the "time to run" area as the current implementation does

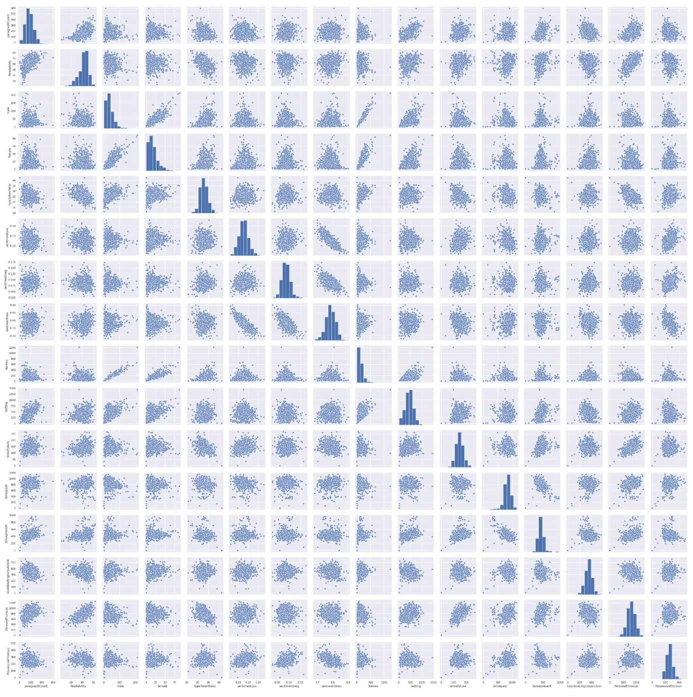
Fig. 4. Pairplot showing extracted features distribution

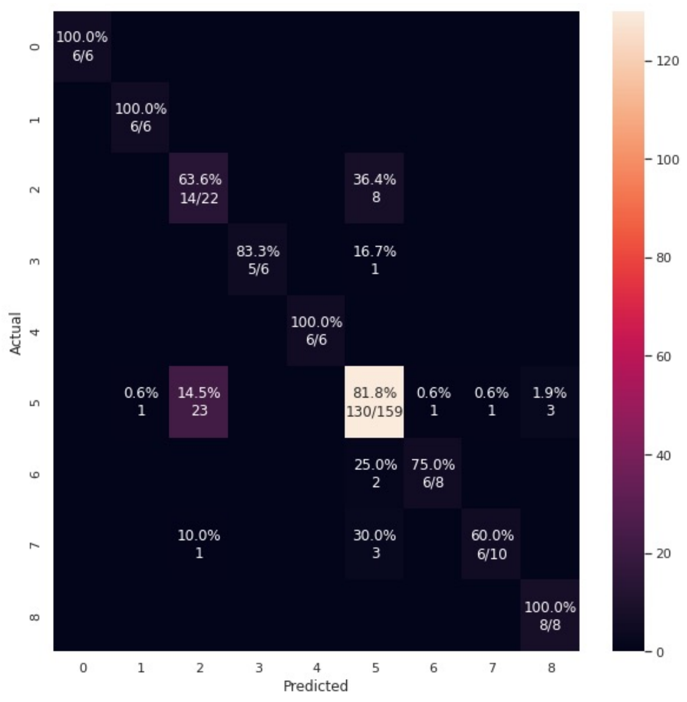

Fig. 6. Confusion matrix for improved distribution

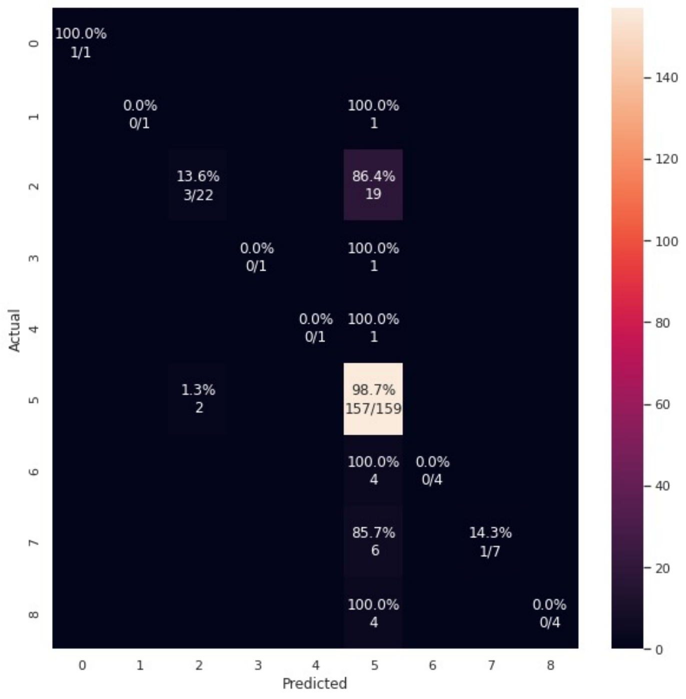

Fig. 5. Confusion matrix for imbalanced class distribution

take a considerable amount of time on a medium sized dataset like Gutenberg corpus and would be at a disadvantage when the model needs to be trained on a larger corpus consisting of many more features and data instances .Hence more research can be carried out on the different ways to lemmatise or compress the data or maybe use various kernel tricks so as to give a more comprehensive understanding of the data or use feature selection techniques like wrapper method ,embedded learning and see whether classifiers trained on these lead to more useful results.

References

[1] S.Polley, S.Ghosh Comparing the qualitative impact of different features and similarities on fictional text using SIMFIC.

[2] S.Polley, S.Ghosh,M.Thiel,M.Kotzbaya,A.Nurnberger SIMFIC: An Explainable Book Search Companion.