Semester Project for Advanced Topics in Machine Learning: Genre Identification of Fiction books from the Gutenberg Corpus

Jayanth Varma Dantuluri
Faculty of Computer Science
Otto von Guericke University
Magdeburg, Germany
jayanth.dantuluri@st.ovgu.de

Pramod kumar Bontha
Faculty of Computer Science
Otto von Guericke University
Magdeburg, Germany
pramod.bontha@st.ovgu.de

SaiSharan Vemu
Faculty of Computer Science
Otto von Guericke University
Magdeburg, Germany
sai.vemu@st.ovgu.de

Devi Prasad Ilapavuluri
Faculty of Computer Science
Otto von Guericke University
Magdeburg, Germany
devi.ilapavuluri@st.ovgu.de

Naveeth Reddy Chitti
Faculty of Computer Science
Otto von Guericke University
Magdeburg, Germany
naveeth.chitti@st.ovgu.de

***Abstract*—Our main goal in this project is to extract features that are relevant to fiction books in Gutenberg corpus subset. We tried to discuss about various features which we extracted based on SIMFIC [1]. We have experimented with training our model based on handcrafted features [1] and evaluated its performance.**
Keywords: Handcrafted features, Genre Identification

I. Motivation and Problem Statement

Reading books is the most common leisure activity for many people irrespective of age.With advancements in technology e-books helps us to access to million of books by a single click.Statistics reveal most of the readers are more inclined towards fiction literature. Genre refers to a category of literature,music or other forms of art based on some set of stylistic criteria.Its estimated that there are about 144 genres and subgenres for fiction literature. Some of the genres in fiction books include Christmas stories,Detective and Mystery,Love and romance and so on. Each genre has some specific attributes which differentiate it. Its a very time consuming task to manually assign a fiction literature to a specific genre. In this project we try to develop a model which will be able to classify a book to a set of genre. We are working on a data set extracted from the Gutenberg project. Our data set comprise of 996 books which belongs to 9 different genres. For classification of books to different genres we would like to explore on some of the low level features and how they help to classify. We would like to answer the below questions

1) Can the selected features of text be used to identify specific genre of fiction literature
2) How is the performance of the model trained on selected features

II. Data set

The data set is subset extracted from Gutenberg corpus. There are 996 books in our data set which is a subset of 19th Century English fiction books. For each book in the data set we have the content in HTML format. The paragraph in the book is enclosed in a ¡p¿ paragraph tag in the HTML document.

The data set is imbalanced as we can see the distribution of books across of the various genres from the figure.1. Out of 996 books 793 books belongs to the genre Literary which accounts to 79.618 percent. Which means blindly if we classify all the books to Literary surprisingly we can get about 79 percent accuracy and hence we should resolve class imbalance problem.

Moreover the genre Detective and Mystery comprise about 111 books. Sea and Adventure genre comprise of 36 books and rest of the 6 genres have below 20 instances.

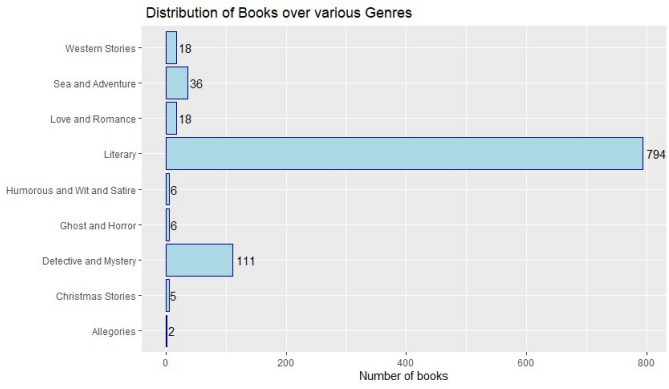

Fig. 1.

III. Concept

Feature Extraction and Feature Scaling methods should be tested.Primarily, Feature Extraction has been done by handcrafting all possible features like Writing Style group, Female

oriented etc. It is a very essential part of our classification problem as, we are dealing with a huge text corpus and obtaining right features which help us to correctly classify genre is desired. We cannot simply extract features using Bag of Words and think it to be effective. We have carried out our feature extraction based on SIMFIC[1]. Based on the notion of similarity, these features are applicable to identify the similarity between the books. There can be various types of similarity like Lexical similarity which is about matching words between books, Semantic similarity which goes about with the similarity in the meaning conveyed by the texts in different books and Syntactic similarity which is about the external structure of the text. Bag of Words simply deals with the presence or absence of particular words and nothing more.

Features like personal pronoun, male pronoun, female pronoun, possessive pronoun, prepositions, conjunctions, paragraph count, semicolon, coma, colon, hyphen, interjection etc are represented by taking their respective count values in every book. As shown in Fig 2 we have represented features and built model based on them. Each of these features helps in making some understanding of the writing style of a type of author. For example, Male pronoun, female pronoun features will typically help to understand the details related to type of characters in the book. For example, some books of a particular author may have many female characters in his/her book. This thing can be common for most of his books and such kind of intriguing features are very important. Another example can be that some authors tend to use more prepositions and punctuations in their books compared to others. Also, use of Interjection or Exclamation more often can be seen in works of authors which talk about lives of criminals.

TABLE I

LIST OF FEATURES EXTRACTED

Feature	Description
Paragraph count	F0
Female Pronoun count	F1
Male Pronoun count	F2
Personal Pronoun count	F3
Possessive Pronoun count	F4
Conjunction count	F5
Prepositions	F6
Comma	F7
Period	F8
Colon	F9
Semicolon	F10
Hyphen	F11
Interjection	F12

Fig. 2.

We used TextBlob library for extracting features and this library depends on NLTK package, used extensively for text processing. Main benefits of using such handcrafted features is that we will enable our model to be trained on various aspects of text which will help the model to recognise some patterns very well and use it for classification. Specifically, these handcrafted features are useful to describe authors' writing styles which are always unique. Like, no two genuine authors' writing styles will look alike. This kind of observation cannot be achieved through Bag of Words approach. On the other hand,one of the problems which we can face using this approach is that we need to rely on assumptions which we choose to make related to the features.

IV. Implementation

Firstly, we read the data from a csv file. The csv file contains details about the bookid, book name, Genre etc. We have used bookid to locate the books in the corpus and parse the books which is in html format using Beautiful soup library. Using the same library we have divide the book into paragraphs. We have used a library called Textblob for accessing the text in the paragraph along with the tags for each word and perform feature extraction from the paragraph level to the book level. So for each book we have extracted 13 features and stored all these features in a csv file.

Once we extract the handcrafted features from the books, taking reference from [1] we are not yet ready to build the model. The Magnitude of the features vary significantly between one feature to the other. Since most of the machine learning algorithms use Euclidean distance for the calculating the distance between two points. It would not be a fair comparison. To overcome this problem, we use feature scaling. As it is a classification problem, standardization is a good way to perform feature scaling. We have used Standardscaler from sklearn framework, it will scale the values in such a way that the mean of the feature is 0 and standard deviation is 1. As a result, all the features are in equal scale. This makes the computation faster and improve model performance.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	0.588020	0.649648	0.226668	1.302164	-0.318939	-0.236550	0.981253	0.610961	-0.195770	-0.262533	-0.182647	-0.476132	-0.389272	1.0
1	-0.315955	-0.365440	0.124985	-0.698457	0.201288	-0.749895	-0.723940	-0.084090	-0.195770	-0.262533	-0.182647	-0.476132	-0.389272	8.0
2	-0.968533	-0.956383	-0.942684	-0.691363	-0.839166	-0.725113	0.279457	-0.257853	-0.294001	-0.262533	-0.182647	-0.476132	1.057425	1.0
3	-1.007950	-0.903960	-1.184181	-0.401037	-1.196822	-0.293195	0.433532	0.263436	0.000690	-0.262533	0.671420	-0.476132	-0.389272	8.0
4	-0.688231	-0.327314	-0.730845	0.107293	-0.839166	0.074997	-0.437255	0.031752	0.197151	2.136394	-0.182647	0.227469	-0.389272	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

Fig. 3. Feature Scaling

After feature scaling is done. We split the data into training and test set where 67 percent of the data is used for training and 33 percent for evaluating the model performance. We also make sure that at least one instance for each class must be in both training and the test set.

For class imbalance problem, we first concatenate training data along with their labels and then split the training data based on their class label. Then we take the count of the majority class and upsampled the minority class with the count of the majority class so that all classes are equally distributed. We have used resample function from the sklearn framework which upsample the instances of the minority class with replacement. After all the minority classes are upsampled, we concatenated the data again and make it as our new training data.

Now, we have training set and the labels ready for model. We have considered the models which works well for multi-class classification. We have experimented with Naive bayes,

SVM and found out that SVM has better performance compared to the Naive bayes. We also thought of to implement Neural network but that requires a lot of training data. We have found that SVM's works well if we have limited training data. It also tries to find out the best fitting line(decision boundary) from our training data. For implementing the machine learning algorithms we have used sklearn framework which contains high-level implementation of several machine learning algorithms.

V. Evaluation

In this we present evaluation and results of the model. Firstly the model for solving the problem is selected by using GridSearchCV module in sklearn framework. GridSearchCV gives the best parameter combination from the given different parameter settings. Then the model with this best parameter setting is trained using training data. The model is then evaluated on the disjoint test data.

We considered different measures for evaluating the model namely precision, accuracy, recall and F1 score which are obtained by Classification report module of sklearn Frame work. The following figure shows the Classification report. Confusion matrix has also been observed which gives number of books per genre are correctly or wrongly classified.

recision	recall	f1-score	support
0.81	0.93	0.87	260
0.36	0.27	0.31	33
1.00	0.12	0.22	8
0.00	0.00	0.00	1
0.00	0.00	0.00	2
0.00	0.00	0.00	0
0.00	0.00	0.00	8
0.00	0.00	0.00	16
0.00	0.00	0.00	1
		0.76	329
0.24	0.15	0.16	329
0.70	0.76	0.72	329

Fig. 4. Classification report

The performance of the model built on handcrafted features is reasonably good. The test accuracy obtained by the model is 76.3 percent. Pre-processing the data, tackling the class imbalance problem, model selection helped us to achieve such results.

Conclusion

Our project main goal is to extract the features relevant to the genre identification on books related to fiction on the Gutenberg Corpus. We totally extracted 13 features related to

```
[[241  13   0   4   0   2   0   0   0]
 [ 23   9   0   0   0   0   1   0   0]
 [  6   1   1   0   0   0   0   0   0]
 [  1   0   0   0   0   0   0   0   0]
 [  1   1   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0]
 [  7   1   0   0   0   0   0   0   0]
 [ 16   0   0   0   0   0   0   0   0]
 [  1   0   0   0   0   0   0   0   0]]
```

Fig. 5. Confusion matrix

writing style such as pronouns, commas and so on., from 996 books and used their genres as the labels. We trained the data on the SVM model and were able to achieve the test accuracy of 76.3 percent. As the accuracy is not the good measure for this text related classification we checked the precision,recall and F1 score and were able to achieve 70, 76 and 72 percent a decent score assuming the retrieved and relevant ratio is good.

But they were haggard tasks we faced during the feature extraction. First, as the corpus was so big and each file had so much text, to extract the features took longer times than we expected. Second, The huge class imbalance problem. Literary books are 79 percent of the corpus. Here we did our best to up sample the other genre examples. We had to leave some features such as sentiment analysis which was taking 10 minutes of time to extract a feature from a single book. This might have contributed more to the classification.

We extracted 13 features from the corpus available. Based on the anova technique, even though we got 5 or 6 best features, they were not alone contributing to the good results. We used all 13 features to train our model to the accuracy of 76.3 percent. As we mentioned above we are able to achieve good scores on precision and recall as well and it infers the classification of the genres is decent.

In the future we would like to add or extract new kinds of features such as semantic analysis, type token ratio, use the different combinations of features for training and get the best feature combination rather than using all features. We would like to understand the relations between each feature and how they are contributing and how they are building up the context. Try these on a base machine learning model like Naive Bayes and explore how they perform on different models.

References

[1] Sayantan Polley and Suhita Gosh, "Comparing the qualitative impact of different features and similarities of fictional text using SIMFIC."