Book Genre Prediction on Project Gutenberg Corpus using Machine Learning

Manali Thakur
Digital Engineering
Otto von Guericke University
Magdeburg, Germany
manali.thakur@st.ovgu.de

Saloni Verma
Data and Knowledge Engineering
Otto von Guericke University
Magdeburg, Germany
saloni.verma@ovgu.de

Shubham Pratap Singh
Data and Knowledge Engineering
Otto von Guericke University
Magdeburg, Germany
shubham.singh@st.ovgu.de

Shweta Bhat
Data and Knowledge Engineering
Otto von Guericke University
Magdeburg, Germany
shweta.bhat@st.ovgu.de

Surabhi Katti
Data and Knowledge Engineering
Otto von Guericke University
Magdeburg, Germany
surabhi.katti@st.ovgu.de

***Abstract*—In this report, we describe our programming assignment for Advanced Topics in Machine Learning where we will create our model for Genre Identification for books using the Gutenberg corpus of books. We outline our approach for selection of features that would be best for genre prediction. We deploy different machine learning models and present our results. They are evaluated on different measures, performance is compared to find the best suitable one. We also show through different graphs and charts for better visualisation of feature distribution and the results.**

***Index Terms*—machine learning, english fiction books, gutenberg corpus, text retrieval, support vector machines, text classification, feature extraction, natural language processing**

I. MOTIVATION AND PROBLEM STATEMENT

A. Introduction

In this report, we outline our undertaking to create our model for Genre Identification for books using the Gutenberg corpus. This project combines the Information Retrieval techniques we have learned so far and the machine learning approaches of prediction, evaluation and visualisation. Using Python as the base for all implementation we propose to build a model for Genre prediction of books.

Genres give the users an idea of the possible content of a book even before having read each title, or its description synopsis. Therefore, predicting the genre is a very strong tool that can sort unmarked text collections or for even searching the world wide web. Relevant problems include summarizing common or combined features shared by different books.

Problem of Interest
Detection of genre of a fiction book using Machine Learning techniques. We extract features from the data that correspond to fiction books like the plot, characters, sentiment analysis to see.

B. Domain

This is a multi-label text classification problem since we are trying to classify books based on a fixed number of features using our model. It falls under the broad category of Natural Language Processing. Classification of books on the basis of subject matter, theme, content has long been a trivial task. Understanding the most important features for a book classification is a time-intensive task, which is understandable and favourable for the user [3].

C. Problem Statement

We present different models for automatic genre detection with a focus on data preparation and modelling obtained from genre and grammatical forms classification of a subset of Gutenberg corpus. Evaluating different models, we choose the best model which outperforms traditional methods often used for text classification.

D. Motivation

In today's world due to ever-increasing demand to make computers perform tasks of humans, machine learning is used. It is a tedious task to manually read the entire book and classify it based on its genre considering the huge number of books available. As said above, manual identification of genre of a book is time-consuming and is difficult to arrange the books in the most convenient order. Classification helps the reader to select the books according to the genre of his choice.

II. UNDERSTANDING THE DATASET

Project Gutenberg [4] is an online library of thousands of free ebooks. For this project, we were given a corpus of books out of which we worked with 996 books in the HTML format and their corresponding 9 labels ie. the Genre. We also have a **csv** sheet including *Book_id, Book_name, guten_genre* and *Author_name*. Each book in the **csv** file has a corresponding HTML file that we need to parse for further use in our machine learning task. The HTML file is the actual book and its content on which we would perform natural language

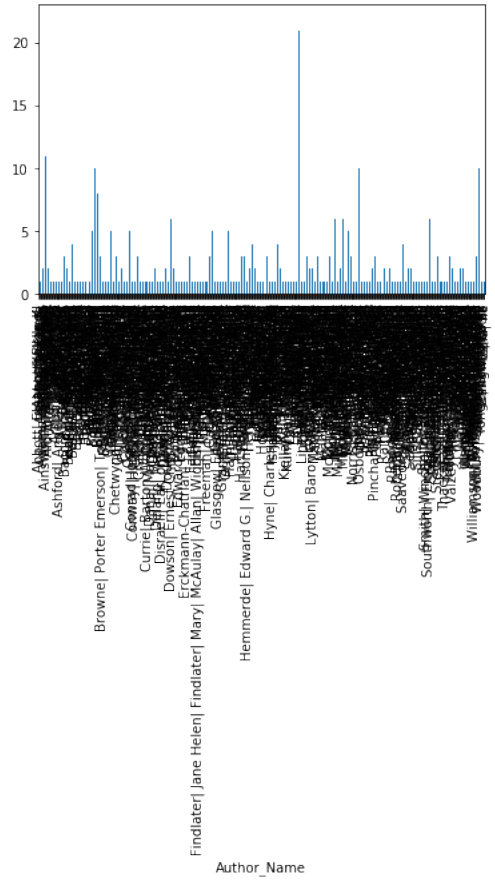


Fig. 1. Initial results of EDA on Author Name and Genre

processing to gather some information. The nine labels consist of Allegories, Literary, Christmas Stories, etc. In fig 1, we can see that the data has severe class imbalance since Literary class contains a high volume of books (794 counts), followed by ’Detective and Mystery’ class (110 counts), steadily declining with Christmas Stories (count 5) and Allegories being almost negligible (count 2). This is one of the issues that are faced in real-world examples because in some models the majority class is favoured. For our task, we split the data in 70-30 proportion with stratified sampling. ie. 70 percent for training our model, and the remaining 30 for testing, ensuring that all the classes get equivalent representation in the data used for modelling and testing. We had also performed some exploratory data analysis before beginning the classification task at hand, to understand how the data was structured and how to proceed. In fig 2, we show one of our initial plots for Author Name and Genre.

III. Concept

The main idea is to first extract features from the dataset given to us in the problem. The elements of fiction - character, plot, point of view, setting, style and themes are a good starting place. We took inspiration from the paper SIMFIC [2] to identify the most useful features for our task.

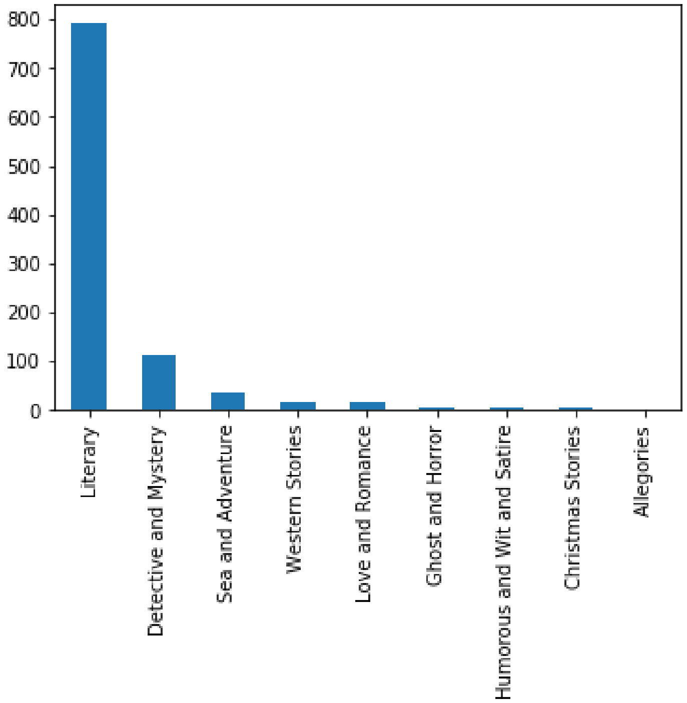


Fig. 2. Bar Chart to show the distribution of classes in the Gutenberg Corpus

In a Bag of Words approach (BoW), it creates a sparse matrix which is very long and may be computationally expensive. It is simple to understand but we miss out on the semantic dependencies and influence of the text documents. So we opt for other approaches that include the semantic meaning of the dataset into consideration for the classification. In this regard, we have represented our features in 3 distinct ways:

- Document to Vector : creates a numeric representation of the document, before running the similarity queries.
- TF-IDF : shows how important a word is to the document in the corpus.
- All features: Taking into consideration all the features, bar none.

We also attempted topic modeling as well using LDA (Latent Dirichlet Allocation), with the feature engineering but the results were not optimum and they are usually non-explainable. So we have not used it for the final comparison document.

We use these feature representation for the next step namely the model learning and testing. We opted for Naive Bayes, Logistic Regression, Support Vector Machines and Neural Networks for our classification. We chose these models because they are simple to implement and their results can be interpreted well. On the other hand, if we choose various Deep Learning models, some approaches give good results but they are not explainable. It works as a black box which becomes complex to explain and interpret for simple problems.

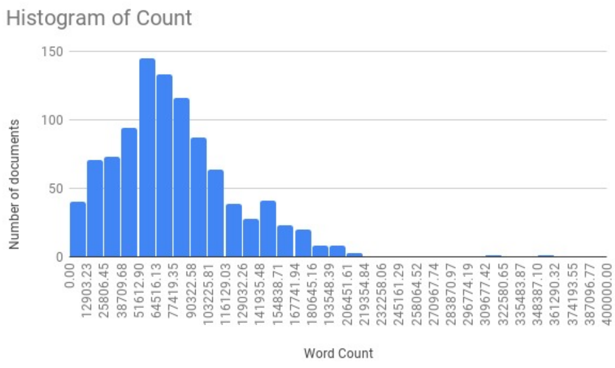


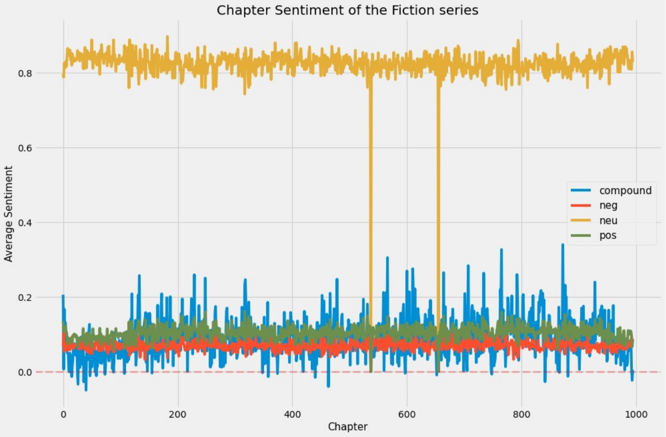
Fig. 3. Histogram of the Word Count against the Number of Documents

IV. Implementation

We deploy various machine learning models to the same problem in the aim to achieve the best results. We compare the performance of these models on the basis of three approaches - testing against all features, document to vector and TF-IDF models.

A. Preprocessing

Preprocessing of raw data plays a vital role in classification task as it directly affects the success rate of the model. In terms of preparing the raw data, we parsed the files and then proceeded with the tokenisation, and conversion of all words to lower case for uniformity across all documents, performed lemmatisation and removed the stopwords (for TTR, doc2vec and TF-IDF). For TTR, another special step was to divide it into chunks. The text is divided into equal sized chunks, so that books with longer length and shorter length are compared fairly. For other parts of Speech features, we did not remove the stopwords otherwise we would have missed the punctuation, quotes that we specifically wanted to focus on. For prelminary EDA we also removed words whose length was less than 3 and removed the remaining tokens that were not alphabetic. Max-Min normalization was used as a feature normalization technique.



.

Fig. 4. Graph of the Book (Chapter on the *x* axis) vs its Average Sentiment. Sentiment values can be Neutral, Positive, Negative and Compound.

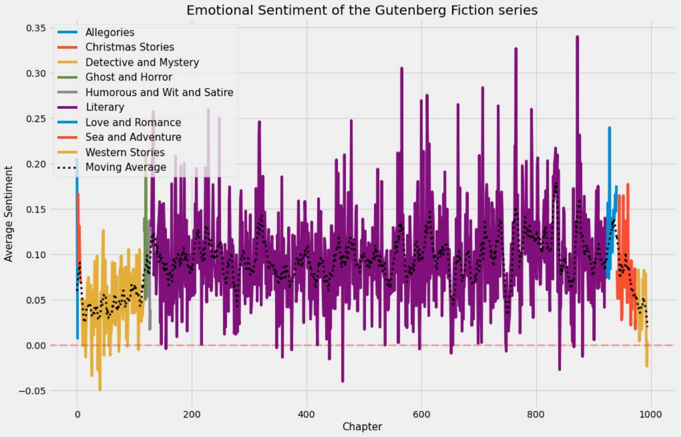


Fig. 5. The Sentiment Plot of the Gutenberg Corpus for all classes (here x: Chapters=Books)

B. Feature Extraction

The next step in the pipeline is the feature extraction. Feature extraction is a process of dimensionality reduction by which a set of raw data is reduced to more manageable groups for processing. The process of feature extraction is useful when you need to reduce the number of resources needed for processing without losing important or relevant information. Feature extraction can also reduce the amount of redundant data for a given analysis. We performed extensive feature engineering in order to find the best representation for a book in terms of inherent characters, writing style, etc. We opted for all following features:

1) Gender Emphasis: Another feature we have engineered was Gender using external Gender Guessers. In this process, we first identified the proper nouns in our text which occurred more than 10 times in our document. We assumed that it would most likely be the name of a person. We then listed the top 100 such words and created their vectors and tried to identify them as male or female using GloVe [8]. It is an unsupervised learning algo used for obtaining vector representations for words. Training was performed on aggregated global word-word co-occurrence statistics from a very large corpus, and the output representations showcase noteworthy linear substructures of the word vector space. Each word hence found is compared in the vector space to find the similarity to the GloVe trained words to predict its Gender. Pronouns like she, he, woman, boy etc are also considered in this count to account for every time the person is not mentioned by name.

- Male count: If negative value, it is a male noun. Since this value is calculated by subtracting value from the female count.
- Female count: Positive value would signify female noun according to the formula. Readers may have a certain affinity for feminist books or male-oriented ones based on their preference.

2) Author: Many readers have an author they gravitate towards. Either because they have read a book by the same author before, or the author comes recommended by a friend or a recommender system online. Users tend to search for a book using the Author's name. This makes it an important feature to be considered. It is considered in our models within the *clean_data* file. Different authors have a unique writing style as well which is a correlated feature detailed further.
3) Plot Complexity: It is another important feature because a plot is considered complex if it has a number of characters, sub-plots, etc. Less characters and one story would be simple to follow, while more characters and and intertwined storylines would be difficult to follow for the average reader.
   - Proper nouns: A good way to measure the plot complexity is by counting the proper nouns in the book. Names can be identified with external libraries that have been trained with proper names. A proper that can occur is that some names are uncommon and may be missing from the library we used. Ex. The name Dudley from the Harry Potter series is uncommon and was found missing from a very small library of common names.
4) Sentence Complexity: Some authors are famously known to write long sentences and coordinating conjunctions. Their punctuations are also pronounced and can alter the reading experience and hence is an important feature from the perspective of a reader.
   - Paragraph Count: This was done by counting the number of paragraph tag elements from the HTML file.
   - Sentence Length: This analysis was achieved without the stopword removal. Readers may choose to read book similar to another one with corresponding sentence lengths - either short or long as per preference.
5) Ease Of Readablilty: Some users prefer to choose books based on ease of readability. They may look for similar books to reading ease as they have read before, or may be looking for a light read next. This is evaluated using the Flesch Reading Score [10] which is given by the formula

   RE = 206.835 – (1.015 x ASL) – (84.6 x ASW)
   where,
   RE = Readability Ease
   ASL = Average Sentence Length (i.e., number of words/number of sentences)
   ASW = Average number of syllables per word (i.e., the number of syllables divided by the number of words)

   This score is a number ranging from 0 to 100. The higher the value of the number, the easier the text will be to read. According to this standard: - Scores of 90.0 and 100.0 : easily understandable by average 5th graders.
   - Scores of 60.0 and 70.0 : easily understood by 8th and 9th graders.
   - Scores of 0.0 and 30.0 : easily understood by college graduates.
6) Writing Complexity The parts of speech consist of Noun, pronoun, verb, adverb, preposition, conjunction, adjective, and interjection. Most authors follow a structure pertaining to the genre of the book. Authors' unique style of writing is called *tics*. The writing style can be deduced by the parts of the speech of the text. A comparison of personal pronouns and punctuations attribute the overall writing style too. Parts of speech were extracted using *nltk* library.
   - Punctuation: Commas, Periods, Colons etc define the pauses. Dashes move us forward, and ellipsis could show uncertainty. They can make us stop to ponder, to move us forward quickly. Colons show the complexity of thoughts where explanation is required. These POS predict the writing style and hence help in classifying the genre also.
   - Double Quotes: This specifies the amount of dialogue and direct speech in a book, found by pattern matching.
   - Preposition-Conjunction: Prepositions like *in, to* etc define the relationships between the subject and object. They describe the position, place and more which are descriptors in our text. Conjunctions like *and, because* are used as connective words. They are used to join different phrases and clauses. Both are main features that affect the overall writing style and also the reading score of that document.
7) Sentiment Analysis is accomplished using Vader, TextBlob NaiveBayes, and Pattern analyzers (Fig 5) and finds the following sentiments:
   - Compound: This is the sum of all lexicon ratings standardised to range between -1 and 1. For example, if a sentence has a rating of 0.67, it will be pretty strongly positive.
   - Negative: If negative words like kill, murder, angry are used in a book it would fall under the negative sentiment.
   - Neutral: No particularly major sentiment could be detected. There is a good balance of positive and negative words.
   - Positive: If words like happy, excited, motivated are used in the book, its general sentiment is regarded as positive.

Sentiment is very important as the reader tries to gauge

TABLE I
COMPARATIVE ANALYSIS OF DIFFERENT IMPLEMENTATIONS, THEIR HYPER-PARAMETERS AND THE VARYING RESULTS

Algorithm	Feature Representaion	HyperParameter	Train accuracy	Test accuracy
G-Naive Bayes	HandCrafted	Default	0.2345323741	0.1872909699
	TF-IDF	Default	1	0.8193979933
M-Naive Bayes	HandCrafted	Default	0.4374100719	0.5150501672
	TF-IDF	Default	0.7971223022	0.7959866221
SVM	HandCrafted	C:1-K:Linear	1	0.845637
		C:0.001-gamma:0.001-K:RBF	0.797122	0.7986577
		C:10-gamma:0.1-K:RBF	1	0.838926
		C:1-gamma-0.001-K:Linear	1	0.845637583
	TF-IDF	C:1-kernel:Linear	1	0.846
		C:0.001-gamma:0.001-RBF	0.797122	0.795986
		C:10-gamma:0.1-RBF	1	0.83946
		C:1-gamma:0.001-Linear	1	0.8461
	DOC2VEC	C:1-K:RBF	0.7985611511	0.7926421405
		C: 0.001-gamma: 0.001-K:RBF	0.798561151	0.79264214
		C:10-gamma:0.01-K:RBF	0.8158273381	0.795986
		C: 0.001-gamma:0.001-K:Linear	0.79856115	0.79264214
Logistic Regression	HandCrafted	solver = 'lbfgs',penalty = 'l2'		0.8993288591
	TF-IDF	solver = 'lbfgs',penalty = 'l2'		0.7993311037
	DOC2VEC	solver = 'lbfgs',penalty = 'l2'		0.7919463087
Neural Networks	HandCrafted	10 unit single layer for 92 epochs	0.972	0.832
	TF-IDF	10 unit single layer for 36 epochs	0.942	0.84
	DOC2VEC	10 unit single layer for 48 epochs	0.869	0.795
	WORD2VEC	2 layer bi-directional LSTM with 100 dimensions per word for 5 epochs	0.797	0.799

the general theme of the book before picking it up from the fiction section. This is also done at a chunk level as the sentiment may vary at different points in the book, and is easier to compare in that way for two books as well. Identifying the sentiment at book level would be a futile task then. Books with similar sentiments can be bundled together.

8) Lexical Richness: Measures the degree of usage of different words, and the number of unique words that have been used in the text. Intuitively, there would be an association with the author and the vocabulary of the book [9]. Some readers want to improve their vocabulary by reading rich books, or find books that match up to their reading levels like for children of different ages.
    - Type Token Ration with chunks: Lexical richness is most commonly calculated with TTR ie. Type Token Ratio. The nearer the value of TTR is to 1, the greater will be the lexical richness of the segment.

$$TTR = \frac{total\ number\ of\ unique\ words}{total\ number\ of\ words}$$

TTR is not the truest representation if the length of the book is very long or short, it becomes less comparable. We use the preprocessed file where the stop words have been removed as well.

C. Feature Selection

Feature Selection aims to choose a small subset of the relevant features from the original ones according to certain relevance evaluation criterion, which usually leads to better learning performance (e.g., higher learning accuracy for classification), lower computational cost, and better model interpretability. We opted to hand-pick some of our feature creating a unique feature space that gave us good results on the models tested. The features mentioned above involved much research and changing up the dataset or model for the best possible results at the time. Chi-Square test and Mutual Information are the two techniques that we could have implemented for this project.

D. Model Selection

Below mentioned are few of the approaches we have chosen in order to implement the classification problem.

1) Multinominal Naive Bayes and Gaussian Naive Bayes
2) Support Vector Machines
3) Logistic Regression
4) Neural Networks

L2 normalisation and default parameters are used for all models. Small changes did not yield significantly different results. All results are in Table 1 and the comparison plots in Fig 9, 10, 11. We can see how different models perform given the 9 class labels for fiction books in our given corpus. We have done one iteration only. SVMs are widely used for the selection of features, but we implemented using different changed hyperparamters(Fig 6, 7, 8) to see its performance and can use it in the future for the purpose of feature selection. We could also do cross-validation as part of future work.

V. EVALUATION

We have selected multiple classification models for the genre classification task. We are evaluating the performance of our selected models on the all handcrafted features, using doc2vec and TF-IDF. The evaluation measures used here for model comparison are precision, recall and f1-score. However, we did not include accuracy, as the data set at hand has a class imbalance problem. This problem might lead to high accuracy score since the prediction might always be the majority class, which might not always be the case. In table 1, we show the comparison results for different models on the basis of different feature representations.

We want to reduce the impact of false positives and hence we chose Precision as our evaluation metric for choosing the best model. Because one could have 100 percent recall yet have a useless model: if the model always outputs a positive prediction, it would have 100 percent recall but be completely uninformative. The models largely perform as expected on the given dataset. A cent by cent metric cannot be achieved when it comes to classifiers due to different assumptions, sampling criteria and other user errors that may result in less than perfect results. Even so, we have achieved reasonable results after modelling. The train-test split incorporated all classes.

‘

A. *Packages and Libraries*

Libraries that were commonly used are: pandas, numpy, sklearn, matplotlib, nltk, Beautiful Soup, pandas, tqdm, matplot, vaderSentiment, and Tensorflow.

B. *Addressing the Class Imbalance Issue*

In our case, we saw that the Literary class in the Fiction Gutenberg dataset had the most number of books. Such problems cannot be handled using standard algorithms since they could be biased towards the majority class. In our case, we used stratified sampling ensuring that at least 1 book from Allegories was part of the test and train split each for equal representation in the dataset. One solution is to artificially augment the dataset by oversampling minority class, undersampling the majority class, or synthesizing new minority classes [5]. It is better to have a classifier that gives high prediction precision or accuracy over the majority class, all the while maintaining reasonable accuracy for the minority classes. Therefore, we leave the classes as it is. A very common solution proposed for this problem is SMOTE (Synthetic Minority Over-sampling Technique) but it is not ideal for text classification since it works on numeric data. SMOTE is ideal for feature space. ie. the output of SMOTE is not ideally a synthetic data which is the real representative of text inside its feature space. Although, SMOTE works with KNN but we know that feature spaces for NLP problems are considerably huge [6] and KNN could easily fail in those higher dimensions.

VI. Conclusion

We can see that all models with the TF-IDF representation perform better than its other two variants namely Doc2vec and hand-crafted features. One reason could be that we opted for Wordnet during our preprocessing which still retains some semantic meaning to our dataset. Feature extraction over emotional anlysis took time over 2 days (completing around 20 percent) and it was discarded due to high computation time. For different models, comparing the precision scores for different feature representations, we concluded that SVM using TF-IDF outperformed other models for this dataset.

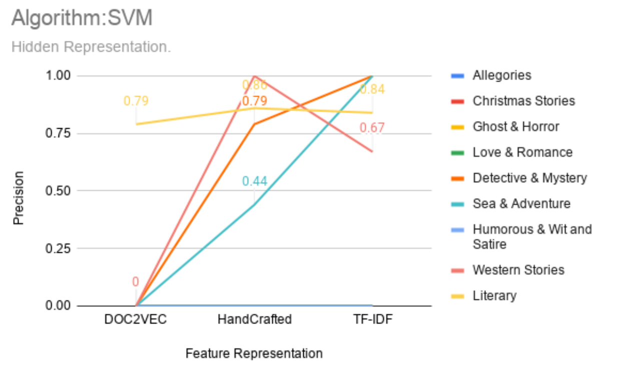


Fig. 6. Support Vector Machine with Different Kernels or Hyperparameters for Hand Crafted Features

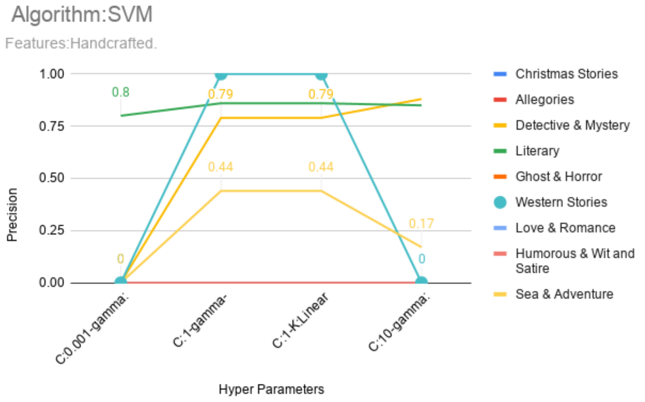


Fig. 7. SVM Visualisation with Alternate Hyperparameters

VII. Further Work

We can try to handle the class imbalance problem using techniques like oversampling the minority class, undersampling the majority class, or synthesizing new minority classes [5]. Another method that works along the basic concept of SMOTE [6]by creating 2 synthetic documents for every original document. Another option could be by implementing SMOTE and ADASYN together with random seed [7]. We can extend the features even more by studying the correlation between them and the classifiers, or by identifying even more appropriate features for fiction books. This scheme should also be generalisable to other genres of book like science, language and art based.

VIII. Related Work

Automatic genre identification is a task which plays a crucial role in many domains such as automatic storytellers, recommender systems and web page topic detectors. Genre classification is especially interesting in the domain of narrative content which is characterized by a large number of

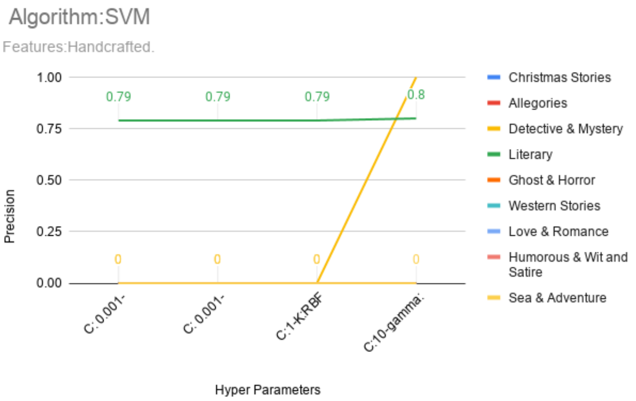


Fig. 8.

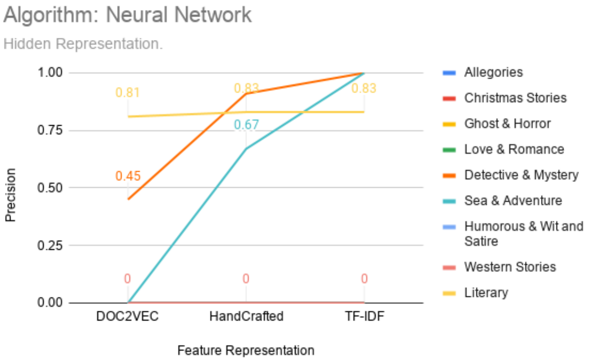


Fig. 9. Neural Network Plot for Precision for all Classes

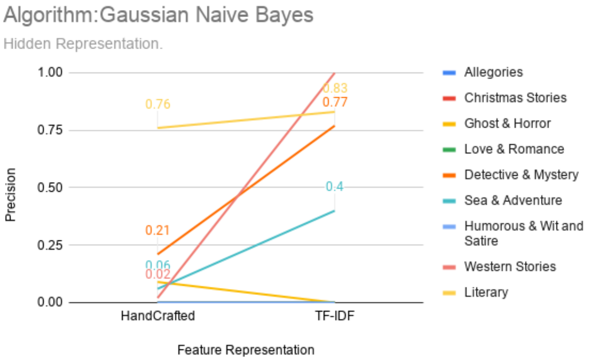


Fig. 10. Gaussian Naive Bayes for Hidden Feature Representation

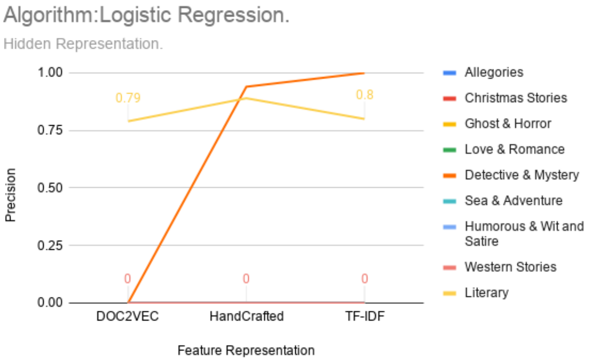


Fig. 11. Logistic Regression Model Plot for Precision for Features with Hidden Representation

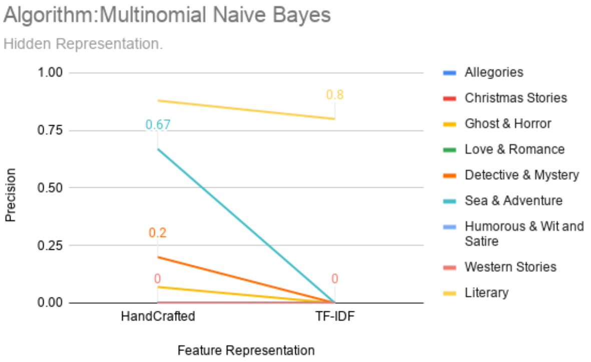


Fig. 12. Multinomial Naive Bayes Graph plotted for Precision

ambiguous and overlapping categories. The rise in popularity of social tagging systems forms a rich source of input information which could be harnessed for this task. A comparison is performed to assess the efficacy of both sources in solving this multi-label classification problem and it is found that the in spite of being expert monitored and better structured, keywords are worse predictors of the genres of movies than tags in most cases. [1]

References

[1] D. Anand, ”Evaluating folksonomy information sources for genre prediction,” 2014 IEEE International Advance Computing Conference (IACC), Gurgaon, 2014, pp. 887-892, doi: 10.1109/IAdCC.2014.6779440.

[2] J. Clerk Maxwell, ”S. Polley, M. Thiel, SIMFIC: An Explainable Book Search Companion”, ICHMS accepted.

[3] Melvil Dewey, Classification and Subject Index for Cataloguing and Arranging the Books and Pamphlets of a Library [Dewey Decimal Classification], Ebook 12513, 2004, Kingsport Press, Inc.

[4] https://www.gutenberg.org/

[5] https://www.kdnuggets.com/2016/08/learning-from-imbalanced-classes.html

[6] https://datascience.stackexchange.com/questions/27671/how-do-you-apply-smote-on-text-classification

[7] https://www.researchgate.net/post/How_can_I_apply_SMOTE_to_text_classification_using_Python

[8] https://nlp.stanford.edu/projects/glove/

[9] Olinghouse, Natalie Wilson, Joshua. (2012). The relationship between vocabulary and writing quality in three genres. Reading and Writing. 26. 10.1007/s11145-012-9392-5.

[10] https://readabilityformulas.com/flesch-reading-ease-readability-formula.php