Development of a Machine Learning Pipeline: A Classification Problem of the Gutenberg Data Set

Felix Böhm, Natalie Nowak, Robert Scharping
Advanced Topics in Machine Learning
Otto von Guericke University
felix.boehm, natalie.nowak, robert.scharping@st.ovgue.de

Marcus Streuber, Jalaj Vora
Advanced Topics in Machine Learning
Otto von Guericke University
marcus.streuber, jalaj.vora@st.ovgue.de

***Abstract*—The project aims to develop and justify a machine learning pipeline for a classification problem regarding to a subset of the Gutenberg data set. The data set is highly imbalanced and is dominated by a seemingly generic class. Those problems are addressed using Random-Over-Sampling and evaluation scores for imbalanced data sets. In the end various classifier are compared with respect to these evaluation measures. The focus lies on feature extraction, model selection and evaluation. The features used are tailored to the domain of fiction, whereby generic word representations are avoided.**

***Index Terms*—Machine Learning, Gutenberg Data-set, Feature Extraction, Text Classification, Class Imbalanced Learning, Multi-Class Classification**

I. Motivation and Problem Statement

The semester task considers a data set with 996 books of 9 genres, whereby the data set constitutes a sub-set of the Project Gutenberg and each book of the data set belonging to the 19th Century English Fiction.

The task is a text classification, where the genre of a book should be established. A special focus of this task was the extraction of features which are relevant for fiction books. For classifying complex data sets like the given one, extracting relevant features is mostly important for accurate results, since especially data sets with long text documents contain a lot of not meaningful information.

Classification tasks like these are relevant for companies like Amazon or Thalia, where each day a bunch of new books are included in the assortment. Having the ability to compare the genre of a book, as well as features of books to similarities, not only allows companies to categorize books more quickly and make book suggestions to customers from the genre, but also to suggest books to customers that share more complex features like sentiment.

Therefore, this work will address the question of how individual features can be extracted from books to make predictions of the genre of not classified books.

II. Data Set

As mentioned, the considered sub-set of the Gutenberg Corpus, the data-set contains 996 books of 9 different genres. Each instance of the data set contains the book name, a book id, the genre and the authors name. The books contents are stored in HTML-files, named according to the book id. Some of these contents contain no text at all or only captions.

The books genre will be the target feature in our task. Table I shows all genres, as well as the number of instances the genre is assigned to. This table points out, that the given data set is highly imbalanced and contains barely any instances for some genres. The imbalance imposes a major problem that needs to be addressed. It's also worth noting, that the most frequent genre 'Literary' seems to be a generic one and doesn't appear in the original Gutenberg data set.

Genre	Assigned Instances
Literary	794
Detective and Mystery	111
Sea and Adventure	36
Western Stories	18
Love and Romance	18
Humorous and Wit and Satire	6
Ghost and Horror	6
Christmas Stories	5
Allegories	2

TABLE I
Genres to Classify

III. Concept

The task is tackled using a standard structure of a text classification pipeline, like shown in Figure 1.

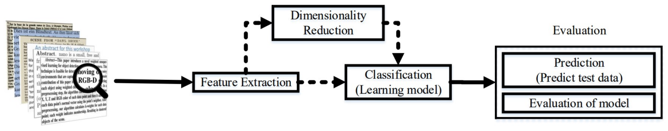

Fig. 1. Text Classification Pipeline [4]

A. Feature Extraction

For the feature extraction a word representation is avoided. Instead, features are extracted that might be relevant for the given task. The main ideas and assumptions, that lead to the features, are described in this section. We decided to represent all feature using numeric values to simplify further processing steps. Due to the low feature count and interest in each features performance, no dimensionality reduction will be performed. Many of the extracted features are inspired by [6].

Title Word Count and Average Word Length: Long and complex or short and simple titles might be typical for a set of genres, as they might appeal to a different audience. Also the use of subtitles increases the title length and could be typical for a genre.

Punctuation Frequency: The frequency of different punctuation marks indicate a certain writing style, which may be typical for a group of authors or a given genre. How the frequency of different punctuation marks may be interpreted is shown in Table II. The given interpretation are guesses and are not backed up by any sources. The frequency of the the punctuation marks is used instead of the count to keep the features comparable between books of different length. The length of a book is accounted for another feature.

Punctuation Mark	Possible Interpretation
.	indicates sentence length, might be different through genres
,	indicates sentence complexity, might be different through genres
”	indicates the use of direct speech $\rightarrow$ vivid stories or plain description
?	indicates the use of questions as a rhetorical tool
!	indicates the use of expressions of excitement or emphasis

TABLE II
PUNCTUATION MARKS AND THEIR INTERPRETATION

Book Length: Long or short books might appear more often in a certain genre.

Proper Names: Proper names in fiction books are probably mainly used for character names and locations. The number of proper names, therefore, might be proportional to the plots complexity. Plot complexity is expected to be different among the given genres.

Point of View: Fiction books use different point of views for different effects. A 1^{st} person narrator is limited to his own thoughts and impressions. This results in tension as information is revealed one piece at a time throughout a book. Next to 1^{st} person narrators there are 3^{rd} person narrators. Those are divided into omniscient and objective narrator. While the first on is aware of all information including the characters thoughts and intents, the second one represent a neutral spectator.

Certain point of views are typically for certain genres. While romances are often written in first person and limited third person view, epics and fantasy are often written from an omniscient 3^{rd} person perspective.

It is assumed that 1^{st} and 3^{rd} person view are reflected by the use of 1^{st} and 3^{rd} single person pronouns. The difference between omniscient and objective narrators are not considered for simplicity.

Sentiment: The sentiment of a book is another feature that could help to identify its genre, because some genre might be identified by a positive sentiment while other are identified by a negative one. Performance on the classification task might also be improved by dividing the books content into multiple sections. This is not exploited for simplicity though. This would result in information about the development of a sentiment. A love story for example might start of positive, fall into a crisis and finally have a happy ending. A story about murder in contrast might start of rather dark and negative. Nonetheless, the sentiment in general is expected to differ between genres.

Content of the books: To analyse the content of a book, the bag of words (BOW) approach seems like an intuitive choice. Here, the words are converted into a numeric representation and can then be further analysed with inverse document frequency to filter the word by rank and get the most important words. But, the outcomes would be mediocre. This is, because it loses many subtleties like context of a word and word ordering. In addition to that, every word would be an additional feature and would overflow our feature collection. After some test runs with the already presented features, in turned out, that only features regarding to punctuation and word length are insufficient to classify a book and we need to implement features regarding to the content.

It is assumed that the books follow a common structure, where the most important part of the book for classification is either the beginning of the book or the climax, which is typically at 4/5 of the book length. A typical curve is presented in 2. Based on this, we decided that the whole book could be represented by the climax. So, we extracted for each book a small sample representing the climax of the book. We can use the climax to compare the books and to get additional features regarding to the content. The next question was, how can we compare those texts and how can we get values for our feature collection to determine the genre. As mentioned, we wanted to avoid BOW due to the fact of a lack of context information. We decided to generate a specific vocabulary for our corpus with words tagged by the genres. For this, the gensim library and the Doc2Vec model is used. The model can then be trained and a similarity function can be used to determine the similarity between the climax of each book and the vocabulary with the tagged words. This results into a score for each book to each genre and focus on the context of words.

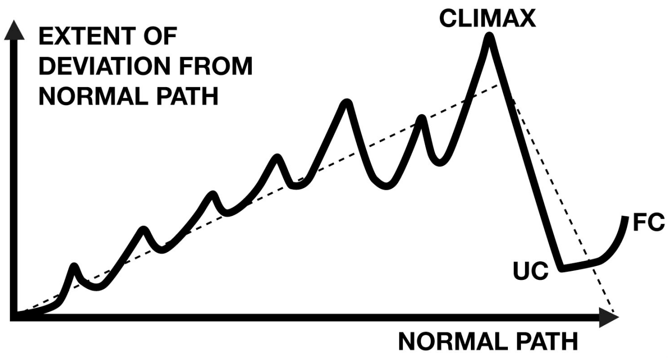

Fig. 2. Fichtean Curve for a Typical Novel [4]

B. Classification Methods

We employ five classification methods to determine the predictive performance of subsets of handcrafted features extracted by a feature-extraction methodology delineated above.

The classifiers are namely Complement Naive Bayes (CNB) [7], Logistic Regression (LR), Random Forest (RF), Support Vector Machines (SVM) [1] and Multi-Layer Perceptron (MLP) [3].

Classifier	Package
Complement Naive Bayes	sklearn.naive_bayes.ComplementNB
Random Forest	sklearn.ensemble.RandomForestClassifier
Neural Network	tf.keras.Sequential
Support Vector Machines with RBF-kernel	sklearn.svm.SVC
Logistic Regression	sklearn.linear_model.LogisticRegression

TABLE III
LIST OF CLASSIFIERS AND PACKAGES USED

We choose specifically these classification methods because they are popular methods w.r.t multi-class target and imbalanced data-set problem. There are other state-of-the-art methods, but in this study we focus on these 5 classifiers. Table III displays the names of the classifiers, and the corresponding package from which the implementation is taken.

To address our class imbalance problem, we decided to use Random Over-Sampling Method to reduce the classifiers bias. We decided on Random Over-Sampling, because [5] implies, that more complex methods this problem won't perform any better. Oversampling was used for all classifiers except the Complement Naive Bayes as it accounts for imbalance by design.

IV. IMPLEMENTATION

Our pipeline was implemented in Python using the nltk library[1] for preprocessing, the genesim library[2] for vector representations of the corpus as well as similarity calculations during the feature extraction and the sklearn[3] and imblearn[4] libraries for training and evaluation.

A. Feature Extraction

Most of the feature extraction, except for the book-content, is performed on string tokens returned by the nltk tokenize function. This representation contains the words as well as punctuation marks.

Title Word Count: The tile word count was calculated by splitting the titles at white spaces and counting them.

Average Word Length: The average word length resulted from splitting the titles at white spaces, determining each tokens length and averaging those.

Punctuation Frequency: The punctuation frequencies were calculated by counting the tokens containing each respective character and then dividing by the books overall token count.

$$\frac{\text{respective punctuation token count}}{\text{token count}} \tag{1}$$

This leads to a feature for each of the five considered punctuation marks. For direct speech only leading quotation marks were considered.

[1]nltk.org
[2]radimrehurek.com/gensim
[3]scikit-learn.org
[4]imbalanced-learn.readthedocs.io

Book Length: For the book length we decided on the overall token count.

Proper Names: The frequency of proper names was calculated by the number of tokens staring with a capital letter minus the ones following a punctuation mark that signals a new sentence. As punctuation mark that signals a new sentence the ones mentioned in Table II except ”,” were considered. The returned number was then divided by the overall word count.

$$\frac{\text{capital word count} - \text{new sentence count}}{\text{overall word count}} \tag{2}$$

Point of View: For the point of view it was assumed that the ratio of the 1st person singular pronoun ”I” to the 3st person singular pronouns ”he” and ”she” represents a measure for the narrator perspective.

$$\frac{\text{"I" count}}{\text{"I" count + "he" count + "she" count}} \tag{3}$$

The higher the resulting number the more the point of view is from a 1st persons perspective. A lower number indicates more use of a 3rd person perspective.

Sentiment The books sentiment was extracted using the nltk VADER implementation. This results in scores for positive, negative and neutral sentiment. The scores add up to 1 for each book.

Content of the books Before the context analysis of the book, some preprocessing of the text was necessary. This includes the elimination of stop words and punctuation. Then, the text is tokenized with a function from nltk and the climax part of the book is extracted. Both full string, as well as the climax string are saved. The full string is from now on referred as the content. The climax is determined as 10% of the book at 4/5 of the length. All contents of the train set is then aggregated by genre. Based on this aggregation, the vocabulary could be defined with the gensim library and trained with the Doc2Vec model. The vocabulary vector size is set to 300 and the model is trained for 30 epochs. With the trained model, the similarity between each of the books climaxes and the genre could be computed and saved as features for the machine learning process. The features are a decimal number that represents the similarity of the genres to each other.

B. Classification Methodology

Complement Naive Bayes was designed to correct the “severe assumptions” made by the standard Multinomial Naive Bayes classifier. It is particularly suited for imbalanced data sets [7]. With the problem in hand, the handcrafted features extracted were initially scaled standardizing features by removing the mean and scaling to unit variance and were given to the classfier. The CNB used has parameters at default setting with α = 1.0.

Logistic Regression also known as logit/MaxEnt classifier. In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme and penalty parameter default to ’l2’ regularisation. The algorithm used in the optimization problem was defaulted to ’lbfgs’ which handles the ’l2’ penalty. All other parameters were used in default. Here again, features scaling were performed to standardize the features before providing to the classifier.

Random Forest is a meta estimator that fits a specific number of decision tree classifiers on various sub-samples of the data-set and uses averaging to improve the predictive accuracy and control over-fitting. The scaling mechanism used here was a bit different, Robust Scaling was used by scaling features using statistics that are robust to outliers as outlier being a severe reason for overfitting in Decision Trees and Algorithms related to it. This ensemble approach was used with all of it parameters being a the default. Due to the over-fitting scenario, the max_depth was set to 5, minimum samples on the leaf was set to 3 and number of estimators taken were 200.

Note, before experimenting with Random Forest Classifier, Decision Trees were implemented just for the sake to understand the possibility of whether a simple Tree based Classifier can deal with such a problem. To this experiment, Decision Trees were trained with max. depth in range (1,50) and an optimal depth of 8 was chosen for training and testing.

Support Vector Machines (SVMs) [2] are a set of supervised learning methods used for classification, regression and outliers detection. When training a SVM with the Radial Basis Function (RBF) kernel, two parameters were considered crucial: C and γ.

The parameter C, common to all SVM kernels, trades off miss-classification of training examples against simplicity of the decision surface. A low C makes the decision surface smooth, while a high C aims at classifying all training examples correctly. Gamma defines how much influence a single training example has. The larger gamma is, the closer other examples must be to be affected. With C and γ spaced exponentially far apart being choosen as good values.

The multiclass support is handled according to a one-vs-one scheme and we selected all the other parameter parameters at default of the tool. To find an optimum and to tune the hyper-parameters C and γ, GirdSearchCV tool from sklearn library was used which being considered crucial for SVM's performance.

Deep Neural Network A NN composed of 4 densely connected layers, with hidden layers using a relu (rectified linear unit) activation function was implemented. The weights of the input layers are initialized within a range of +/- 0.1, and the learn rate is set to 0.005, while the hidden-layer-size varies from 128 to 32.

V. Evaluation

A. Feature Extraction Evaluation

Looking at our features and their respective histograms and scatter plots, we could identify the following properties:

- As stated in previously in Section III-A we agreed on numerical values for our features.
- Except for the word counts and word length all features are in an interval between 0 and 1.
- Looking at the histograms of each respective feature, most distributions seem to follow a multinomial distribution. The others seem follow a normal distribution.
- Significant positive correlations could be identified between the use of quotation marks and question marks, as questions often used within direct speech.
- Negative correlations was identified between different sentiments. This results from the fact, that positive, negative and neutral sentiment values always add up to 1.

B. Classifier Performance Evaluation

Given the highly imbalanced data set, it must be ensured that every genre we want to be able to predict, is also accurately represented in our test set. Therefore, we use a stratified split for the training's and test data, which will result in both sub sets representing genres in the same proportions. This will still mean that our test set is imbalanced though, so a normal accuracy calculation will not be sufficient, since this will result in great performance of models that always just predict the over represented 'Literary' genre. For this case, sklearn has a balanced accuracy score function, which will weight scores according to their genres occurrence count, and can even take an additional parameter to modify the result in respect to chance, so a random performance will result in a 0. We compared some of our implemented classifiers performance using these accuracy measures as seen in Table IV.

For our evaluation we discarded instances that belong to

Classifier	Balanced Accuracy in %	Adjusted in %
Complement Naive Bayes	15.2	1.0
Decision Trees	31.2	19.7
Random Forest	44.6	34.4
Neural Network	49.4	40.9
Support Vector Machine	39.9	29.9
Logistic Regression	35.9	25.3

TABLE IV
BALANCED AND ADJUSTED-FOR-CHANCE ACCURACY SCORES

'Allegories' or 'Christmas Stories', since our data set does not have enough unique instances to represent them in a learn-able fashion. Though the next 2 lowest represented genres were also not learn-able for some classifiers, which can be seen in Figure 3, where the SVM never classifies anything as 'Horror' or 'Satire' for the test set, however they are being represented in it.

One of the main reasons for a poor performance in this task is the given data set. The 'Literary' genre has an overlap with the other genres since books of it are simply relabeled instances of other classes. So as a 'Detective' novel may be labeled 'Literary' in the data set, an algorithm that actually learns what a 'Detective' novel is will perform poorly, since in the test set, some of them are classified as Literary. Similarly learning a model on such a data set will have difficulties since similarity measures for a specific genre will also be likely similar to 'Literary'. This overlap can be seen in the confusion matrix in Figure 3, since almost every missclassified instance was either 'Literary', or predicted as 'Literary'. Thus the learned

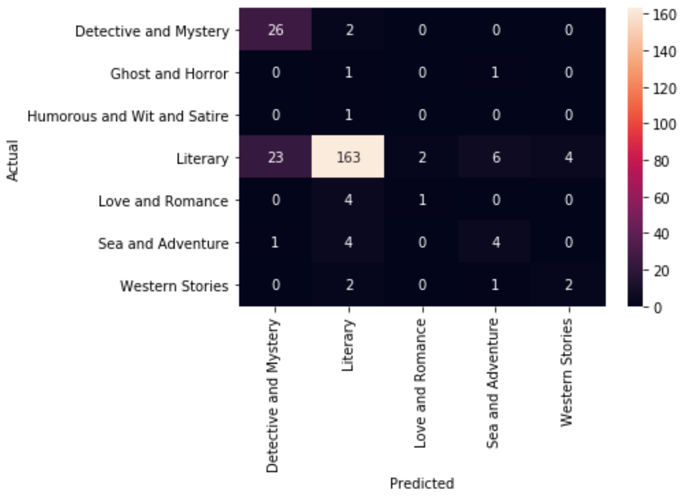

Fig. 3. SVM confusion matrix heat map

models have trouble telling 'Literary' apart from all the other genre.
For the learning of a model, this 'Literary' genre can also not be ignored, since otherwise our data-set would be too small to learn anything. In reality the 'Literary' genre is not a real genre of 'Fiction books', so it is not a valid label in regard to our domain knowledge. Though it could have been used as unlabeled data, we got the impression that in the scope of the task, 'Literary' was supposed to be a valid genre, that was added to artificially introduce an imbalanced data problem.

VI. Conclusion

As part of the semester assignment a classification pipeline was developed. Considering the balanced accuracy score, the deployed models performs rather badly in comparison with other classification tasks. This could be due to several reasons. First, the data set imposes hard to address challenges. On the one hand, the data set was highly imbalanced with 794 instances representing the most frequent genre and 2 instances representing the least frequent classes. On the other hand, the most frequent class "Literary" seems to be generic. Therefore, the data set seems to contain more than 75% distracting, meaningless data. Last but not least, the low experience of the authors probably lead to some bad assumptions and design choices.

Looking back, there are a few points we could improve on:

- the choice of classifiers could be altered
- the classifiers hyperparameters could be further tuned to enhance their performance
- additional meaningful features could be added
- dropping the literary label and using the data for an semi-supervised learning approach could be considered

References

[1] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[2] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

[3] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. The MIT Press, 2016.

[4] K. Kowsari, K. J. Meimandi, M. Heidarysafa, S. Mendu, and D. Barnes, L.and Brown. Text Classification Algorithms: A Survey. Technical report, Department of Systems and Information Engineering, University of Virginia, 2019.

[5] Mateusz Lango. Tackling the problem of class imbalance in multi-class sentiment classification: An experimental study. *Foundations of Computing and Decision Sciences*, 44(2):181–178, 2019.

[6] Sayantan Polley, Suhita Ghosh, Marcus Thiel, Michael Kotzyba, and Andreas Nurnberger. *SIMFIC: An Explainable Book Search Companion*. 2020.

[7] Jason DM Rennie, Lawrence Shih, Jaime Teevan, and David R Karger. Tackling the poor assumptions of naive bayes text classifiers in: Proceedings of the twentieth international conference on machine learning. *Whasington DC*, 2003.