

Classification Assignment 1

Data Science Applications DTI5125[EG]

Group name: DSA_202101_ 8

Amr Ashraf Mahmoud Elsherbiny

Ahmed Fares Saad Eldin Khalifa

Amira EssamEldin Ibrahim Ibrah Elgebaly

Abdallah Medhat Mohamed Rashed

Contents

1. Objectives	2
2. Requirements	2
3. Methodology	2
3.1. Data Preparation.....	2
3.1. Data Exploration	2
3.2. Data Preprocessing.....	4
3.3. Data Transformation	4
a) Bag of Words	4
b) Bag of Bigrams	6
c) Term Frequency-Inverse Document Frequency.....	6
d) Doc2Vec	7
e) Latent Dirichlet Allocation	8
e) LDA + TF-IDF.....	9
3.4. Champion Model	9
3.5. Error Analysis	9

Figures

Figure 1: Reading books and generate partitions of 100 words.....	2
Figure 2:Sample of Clean Words.....	3
Figure 3: Adjective features per books	3
Figure 4:Noun features per books	3
Figure 5: Avg word length per book.....	4
Figure 6: Char per book	4
Figure 7: Punctuation count per book	4
Figure 8: most frequent 20 words in book after BOW.....	5
Figure 9: model accuracy with 200 max features in BOW	5
Figure 10: model accuracy with 500 max features in BOW	5
Figure 11:max features in BOW	6
Figure 12: most frequent words in BOW 2-gram	6
Figure 13: most frequent words in TF-IDF.....	7
Figure 14: model accuracy with 200 max features in TF-IDF.....	7
Figure 15: model accuracy with 500 max features in TF-IDF.....	7
Figure 16:most frequent words of max feature in doc2vec.....	8
Figure 17 dominant topics in books and classifier accuracy.....	8
Figure 18:models on combination TF-IDF and LDA	9
Figure 19:comparison between model performance in different transformation.....	9
Figure 20:confusion matrix on champion model	10
Figure 21:Misclassifications per Class	10
Figure 22:plotting error prediction from champion model	10

1. Objectives

The overall objective is to analyze similarities between partitions of different books from the same genre, use different transformations and models to classify these partitions, analyze the pros and cons of algorithms, and finally generate and communicate insights.

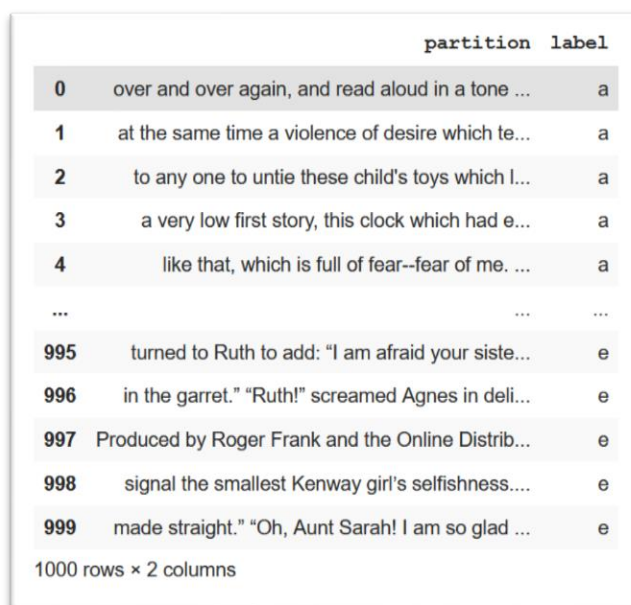
2. Requirements

Using a random sample of size 5 of similar books written by different authors from Gutenberg Digital Library, split the data into training and testing partitions. After that, try different transformation techniques and classification methods to accurately classify these partitions.

3. Methodology

3.1. Data Preparation

We start by loading the books from URLs in Gutenberg library website. After that, we proceed by removing the header and footer of Gutenberg library from each book. We then create random samples of 200 partitions from each book. As we sample before cleaning to avoid losing important features, we prepare the records of 200 words records for each document, label them as a, b and c etc. as per the book they belong to. Figure 1 shows a sample of the resulting dataframe.



	partition	label
0	over and over again, and read aloud in a tone ...	a
1	at the same time a violence of desire which te...	a
2	to any one to untie these child's toys which l...	a
3	a very low first story, this clock which had e...	a
4	like that, which is full of fear--fear of me. ...	a
...
995	turned to Ruth to add: "I am afraid your siste...	e
996	in the garret." "Ruth!" screamed Agnes in deli...	e
997	Produced by Roger Frank and the Online Distrib...	e
998	signal the smallest Kenway girl's selfishness....	e
999	made straight." "Oh, Aunt Sarah! I am so glad ...	e

1000 rows x 2 columns

Figure 1: Reading books and generate partitions of 100 words

3.2. Data Exploration

After loading and sampling from the 5 books, we explore various features of the raw data to get a better understanding of the structure of each book and the author's writing style. First of all, we plot the word counts of each book, as shown in figure 2, to understand the common

words in each one. We observe that many of the common words are names that are unique to each book, like main characters, which we will have to deal with later. We then employ a part-of-speech tagger to tag each author's sampled partitions. Figures 3 and 4 show density plots of different parts of speech, grouped by book. In addition to that, we extract text features like average sentence length and amount of punctuation used, and plot their densities as shown in figures 5-7. To assess whether these features are valuable in discriminating between authors, we train a simple random forest and plot the feature importance.



Figure 2: Sample of Clean Words

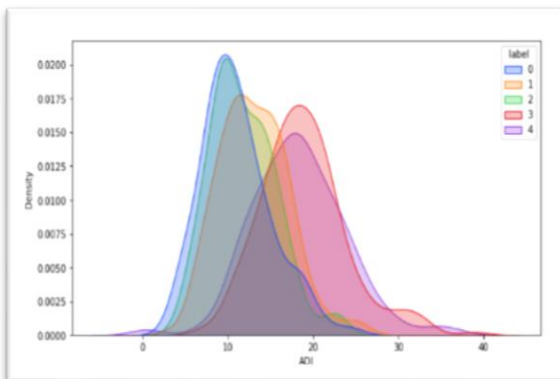


Figure 3: Adjective features per books

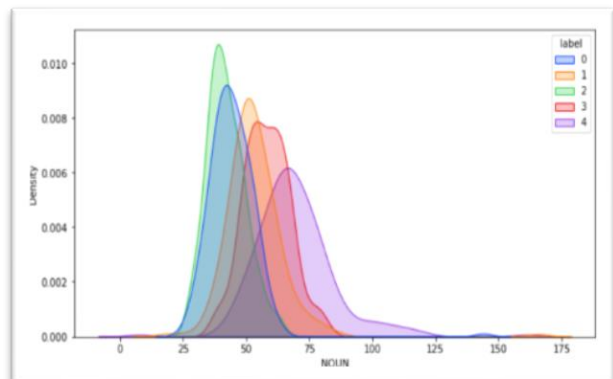


Figure 4: Noun features per books

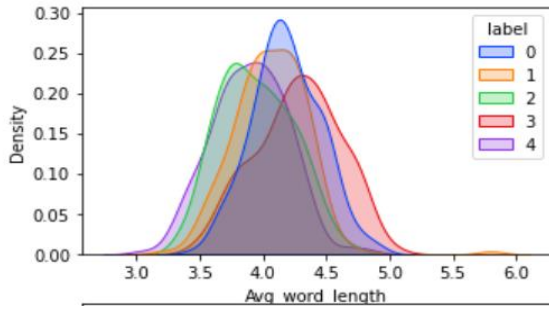


Figure 5: Avg word length per book

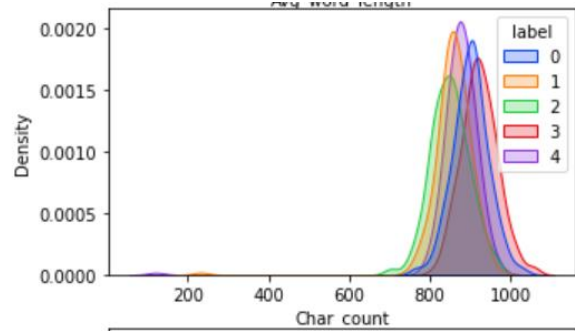


Figure 6: Char per book

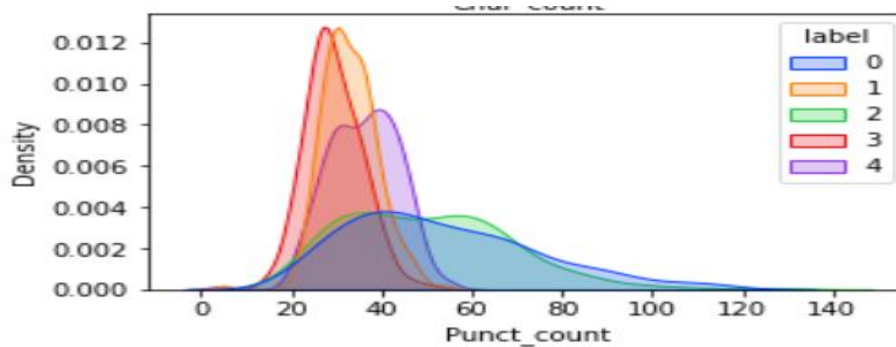


Figure 7: Punctuation count per book

3.3. Data Preprocessing

We start by removing stop words and punctuations by regular expression, which are the noise in the text. In addition to that, we normalize text to words by using Stemming which reduces words to their word root. Finally, we remove people names from every book which may conflict our models and know the book from the names which used on it.

3.4. Data Transformation

Data Transformation is important because in the problem of classification of texts, we have a series of texts and their corresponding labels. But we directly can't use text for our model. We have to convert that text into some numbers or number vectors. And we applied algorithms on every transformation model with variant features which affect the accuracy, so when we choose low number of max features we get low accuracy.

a) Bag of Words

The first transformation we apply is bag of words. It basically runs over the whole dataset, constructs its vocabulary of predefined size, and then transforms each sentence to number of occurrences of each word in it. To account for varying size sentence, we normalize the vector by the number of words in the sentence. After that, we pass the transformed vector to 3 different classifiers, namely Multinomial Naïve Bayes, Random

Forest, and Support Vector Machine. Figure 8 shows the 20 most frequent words of a book after transformation. Figures 10 and 11 show the classifiers' accuracy, trained on bag of words transformed vectors, with vocabulary size of 200 and 500, respectively. To inspect the effect of increasing the vocabulary size, we vary the size of the vocabulary and train our classifiers. Figure 11 shows the increase in accuracy by increasing the vocabulary size.

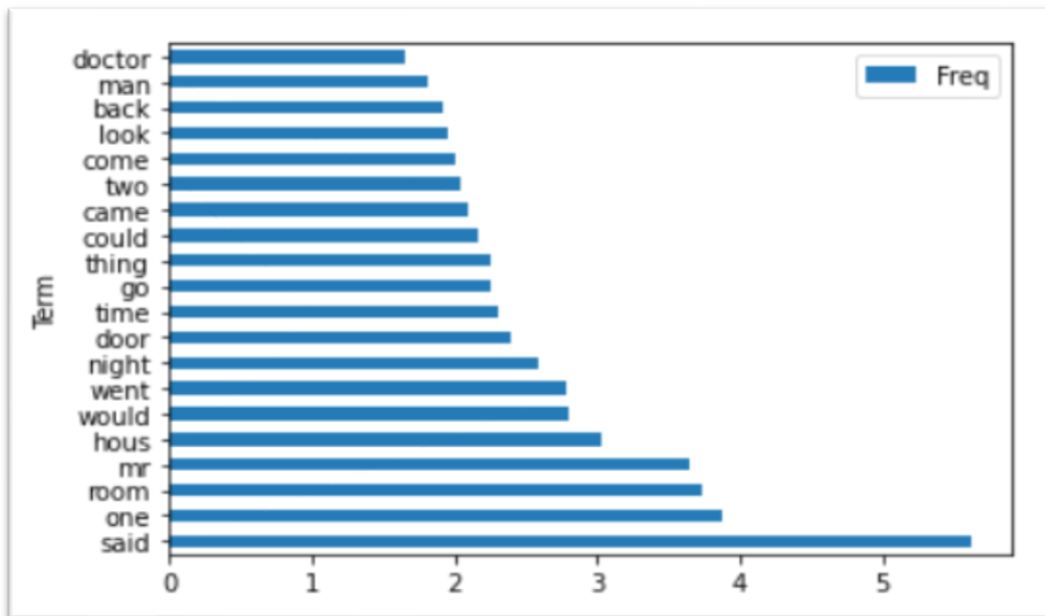


Figure 8: most frequent 20 words in book after BOW

```

----- MultinomialNB -----
kf scores: 60.750 %
----- RandomForestClassifier --
kf scores: 75.000 %
----- SVC -----
kf scores: 78.375 %

```

Figure 9: model accuracy with 200 max features in BOW

```

----- Fetures = 500.0 -----
----- MultinomialNB -----
kf scores: 58.000 %
----- RandomForestClassifier --
kf scores: 82.500 %
----- SVC -----
kf scores: 88.000 %

```

Figure 10: model accuracy with 500 max features in BOW

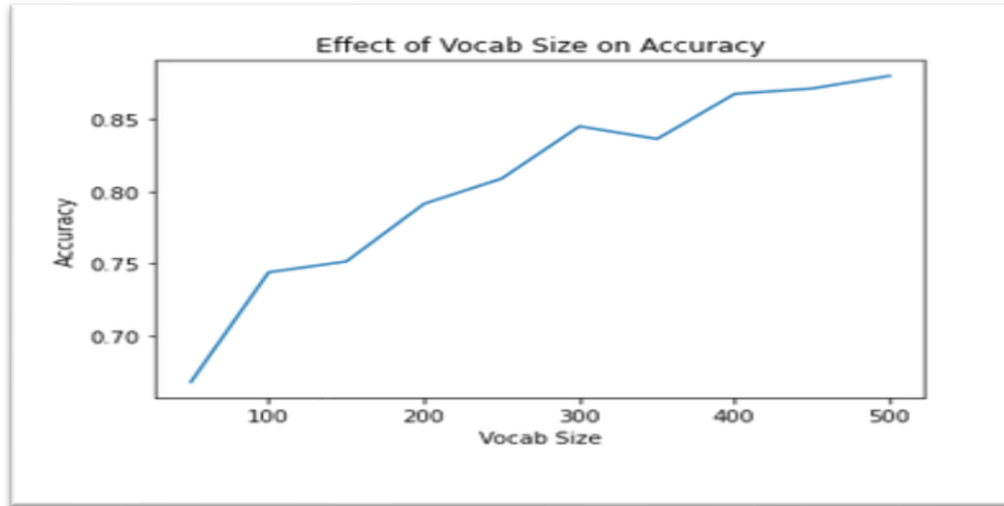


Figure 11: max features in BOW

b) Bag of Bigrams

Aiming to capture some of the sentence structure in the books, we use the same Bag of Words approach mentioned in the previous section, but this time with each 2 consecutive words. Figure 12 shows the 20 most frequent bigrams in a certain book. As the number of possibilities significantly increase by this method, choosing the right vocabulary size becomes challenging.

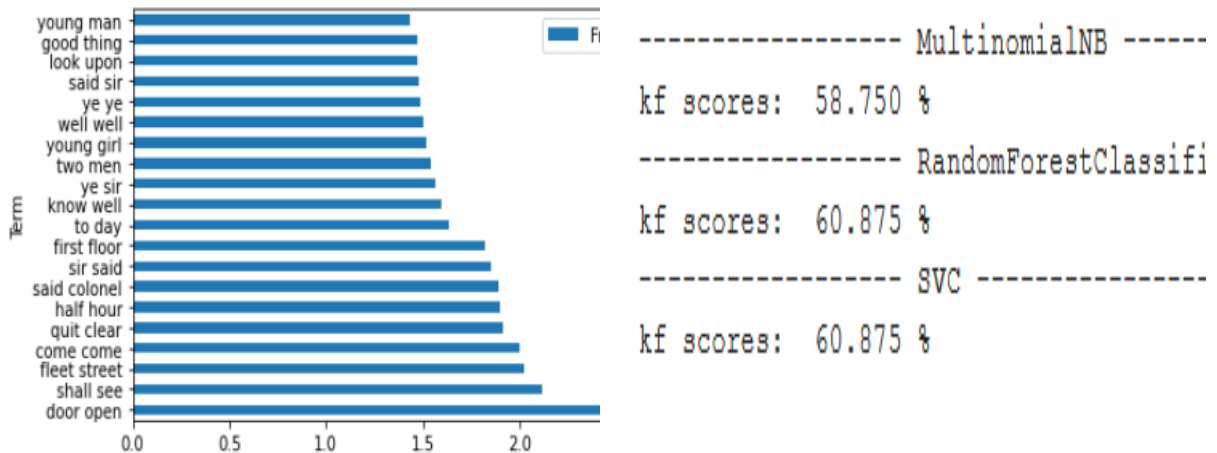


Figure 12: most frequent words in BOW 2-gram

c) Term Frequency-Inverse Document Frequency

A clear drawback of the simple Bag of Words transformation is that it does not consider very common words that are present in many documents. These frequent words are likely to dominate the vocabulary, carrying little information about the differences between documents. To address that, we use Term Frequency-Inverse Document Frequency

transformation, which assigns a lower weight to the too frequent words. Figure 13 shows the 20 most frequent words after scaling.

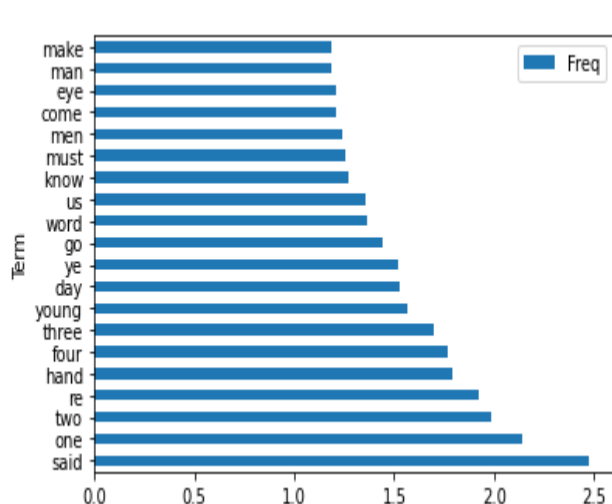
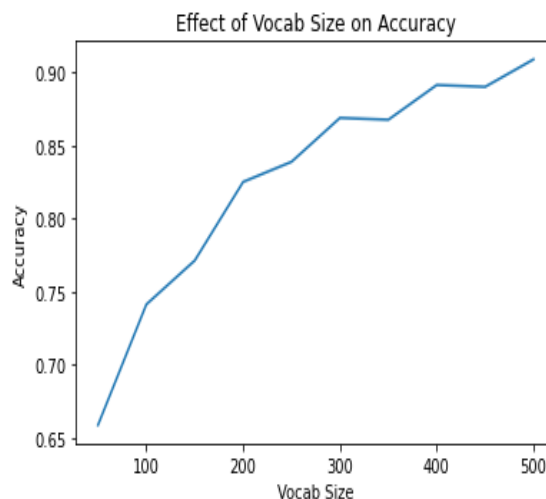


Figure 13: most frequent words in TF-IDF



```
----- MultinomialNB -----
kf scores: 59.125 %
----- RandomForestClassifier --
kf scores: 78.000 %
----- SVC -----
kf scores: 82.500 %
```

Figure 14: model accuracy with 200 max features in TF-IDF

```
----- Features = 500.0 -----
----- MultinomialNB -----
kf scores: 50.125 %
----- RandomForestClassifier --
kf scores: 83.250 %
----- SVC -----
kf scores: 90.875 %
```

Figure 15: model accuracy with 500 max features in TF-IDF

d) Doc2Vec

A more sophisticated transformation that we use here is Doc2Vec. It represents the whole partition of text as a fixed length vector, and allows computing similarities. To find the best vector size, we vary it and train different classifiers and keep track of the performance.

Figure 19 shows the effect of varying the vector size on classification accuracy.

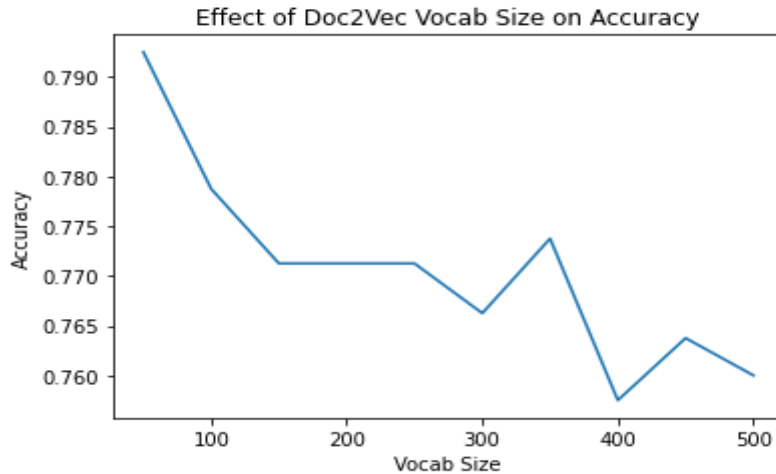


Figure 16: most frequent words of max feature in doc2vec

e) Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) assumes that a document is a mixture of topics, and topics are mixture of words. To find the best number of topics for our data, we use grid search over different number of topics. After doing this, we found out that the best number of topics is 3. Figure 20 shows the distribution of the dominant topics in each book and the classifiers' accuracies. While there seems to be differences between the topics, it is not enough to accurately classify the documents as shown in the Random Forest and SVC low accuracies.



Figure 17 dominant topics in books and classifier accuracy

e) LDA + TF-IDF

LDA and TF-IDF capture different information, therefore combining them may increase the classifiers' ability to accurately predict the author. Figure 21 shows the accuracies of Random Forest and SVC classifiers, trained on combined LDA and TF-IDF.

```
----- RandomForestClassifier
kf scores: 84.250 %
----- SVC -----
kf scores: 85.875 %
0.85875
```

Figure 18:models on combination TF-IDF and LDA

3.5. Champion Model

TF-IDF alone and TF-IDF LDA combination seem to perform similarly when use to train SVC on cross validation data, therefore we use the test data to score them. By doing this, we see that combining TF-IDF and LDA actually helped improve the prediction capabilities, as it scores 85.7% on test data, compared to 81% by TF-IDF alone. We therefore choose the TF-IDF LDA combination along with the SVC to be out champion model.

Model (Features)	Multinomial NB	Random Forest Classifier	SVC
BOW(300)	63.125 %	80.625 %	85.000 %
BOW 2-gram(200)	60.750 %	63.000 %	63.625 %
TFIDF(300)	58.375 %	81.250 %	86.875 %
TFIDF n gram(200)	60.625 %	66.125 %	64.000 %
Doc2Vec(50)	-	77.875 %	79.250 %
LDA(200)	-	49.750 %	58.000 %
Combining TFIDF with LDA (300)	-	84.250 %	85.875 %

Figure 19:comparison between model performance in different transformation

3.6. Error Analysis

Figure 20 shows that book number 4 was the most accurately classified boo, while book 0 was the most commonly misclassified book. Using LDA model to predict the dominant topic in the misclassified examples, figure 21 shows the dominant topic in each misclassified

example, grouped by the true label. While there does not seem to be a pattern here, it is worth noting that in book number 3, the dominant topic was topic 2, and all misclassifications were from topics 1 and 2.

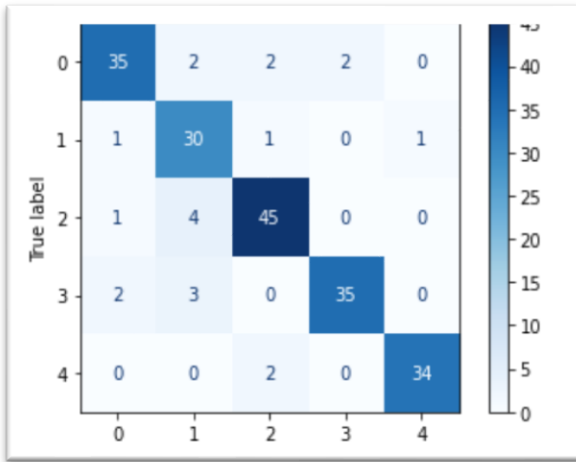


Figure 20: confusion matrix on champion model

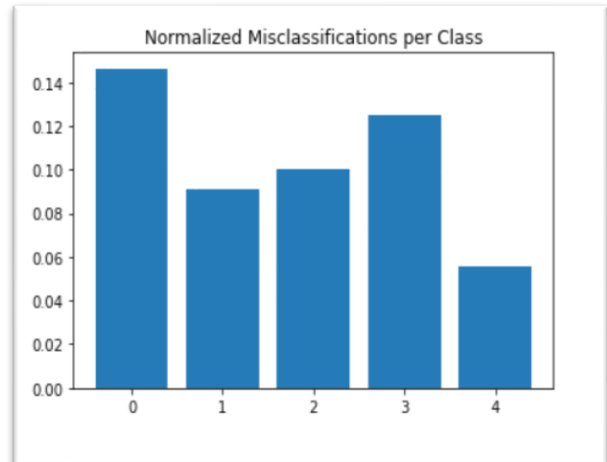


Figure 21: Misclassifications per Class

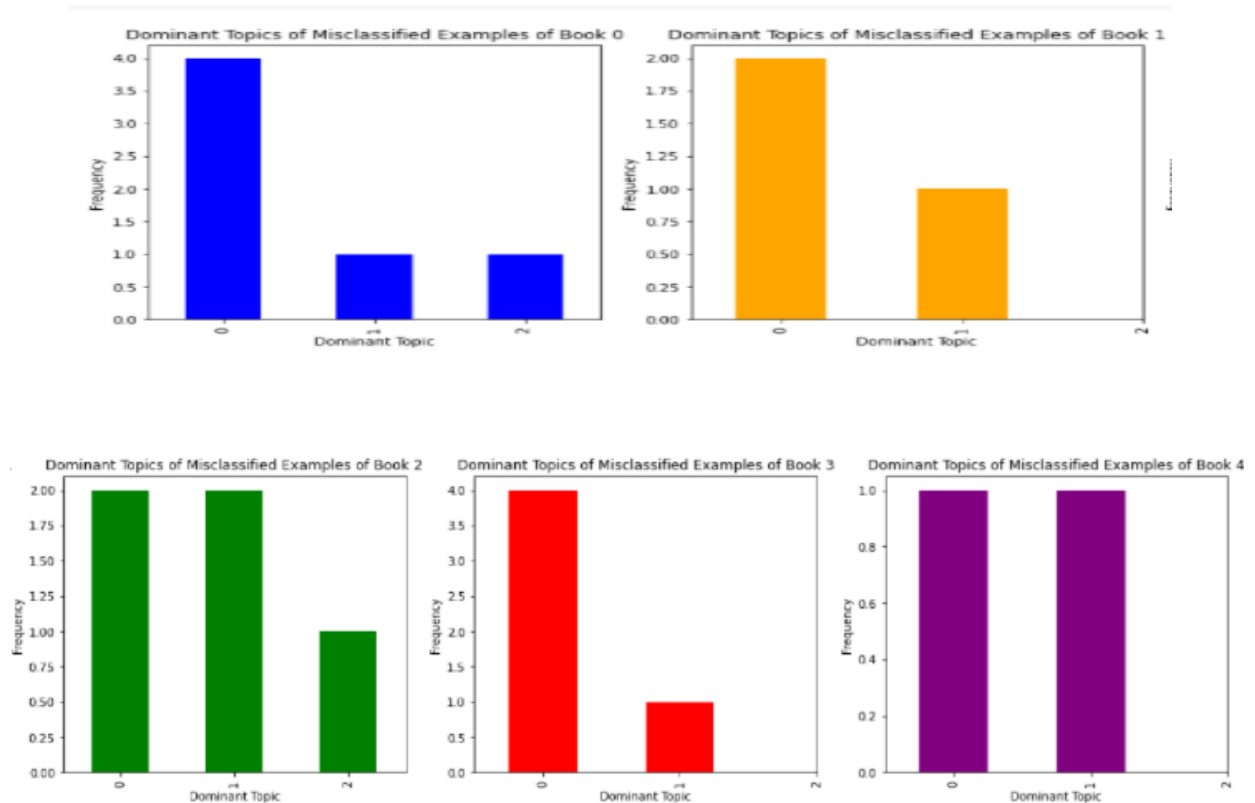


Figure 22: plotting error prediction from champion model