University of Ottawa

Department of Electrical and Computer Engineering

GNG 5125/ Data Science Applications

Professor: Arya Rahgozar

Project Report on Text Classification

Submitted by

Group: DSA_202101_ 7

Yusri Al-Sanaani	300216450
Hetvi Soni	300200976
Tavleen Kour	300213090
Immanuella Iyawe	300150838

Due Date: Monday, 8th January 2021

Introduction

In this report, three machine learning algorithms are used to classify text data taken from different books with different authors of Gutenberg digital books. The overall objective is to produce classification predictions and compare them; analyze the pros and cons of algorithms and choose the best model. The confusion matrix, classification report, accuracy, bias, variance, and MSE are used to evaluate the performance of models on both training and testing data.

1. Text Data Preparation and Pre-processing

Data set used for this project was obtained from Gutenberg's Digital library. Five different books with different authors have been imported and then pre-processed. Data preparation, data pre-processing, and text analysis will be discussed in this section.

1.1. Data preparation

For imported books, each book has been partitioned into chunks based on 100 words each, and each chunk was labelled to the book it belongs to. Then a set of 200 chunks has been randomly chosen from each book to create unbiased data set. The data set was stored in a data frame. We can summarise these steps as follows:

- Import five books.
- Extract the book titles and author names using regex to get the data frame shown in Table 1:

Table 1:Books data frame.

	Books	Authors
1	The Man Who Was Thursday	Chesterton
2	Alice's Adventures in Wonderland	Carroll
3	Leaves of Grass	Whitman
4	Paradise Lost	Milton
5	Stories to Tell to Children	Bryant

- Partition each book to chunks (100 words for each chunk); a function has been defined to partition multiple books (get_chunks). This function takes dictionary as input that contains the labels (books and authors names) and books content. It returns a list of partitioned books with labelling each partition to the book it belongs to.
- Choose 200 random chunks from each book by defining a function (get_random_chunks) to return 200 random chunks for each book. This function takes a list of previously partitioned books as input. It returns a list of books containing 200 random chunks each.
- Store the obtained data set in a DataFrame in the form shown in Table 2.

Table 2:The shape of the data frame

	Books text	Authors labels
0 . . 999	. . Text data . .	. . Authors . .

1.2. Data Pre-Processing

After preparing the data set and storing it in a data frame as shown in Table 2 and figure 2, further steps have been performed to clean the data set, perform lemmatization and stemming, and remove stop words. Figure 1 summarizes the pre-processing procedure to prepare the text data for feature engineering.

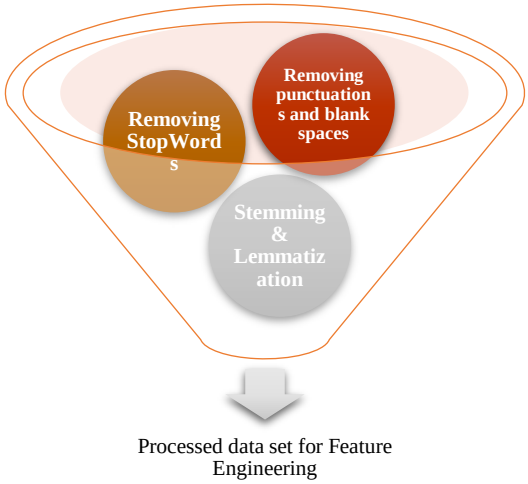

Figure 1: Data pre-processing process.

1.2.1. Data Cleaning

The data cleaning was done by creating a general function (get_clean_text) to clean the data of multiple books. This function takes a list of books as input and returns a list of cleaned books. The cleaning includes:

- Removing multiple spaces and left/right white spaces.
- Converting capital characters to lower case.
- Removing special characters.
- Removing single character word.

Then, the cleaned data set was stored in data frame as shown in figure 3.

	books_text_data	authors_labels
894	the War-dance . The Woodpecker came and sounde...	Bryant
45	like a silver flame on his forehead . For a lo...	Chesterton
478	the pavements , merge with the crowd , and gaz...	Whitman
387	coming down ! Heads below ! ' (a loud crash)...	Carroll
698	whom they wished beheld , Their mighty Chief r...	Milton

Figure 2: DataFrame for the uncleaned data set.

	books_text_data	authors_labels
516	you and henceforth possess you to myself and w...	Whitman
970	you would not help me and burned hotter than e...	Bryant
332	that it was a very difficult game indeed the p...	Carroll
709	me according to his will therefore to me their...	Milton
575	in the temple or heart the living and dead lay...	Whitman

Figure 3: DataFrame for the cleaned data set.

1.2.2. Stop Words Removal

The data samples from the book are then prepared for further process using nltk library to remove the stop words. Figure 4 shows the data frame for the data set after removing stop words.

	books_text_data	authors_labels	text_no_stopwords
568	woodpile through the swung half door of the ki...	Whitman	woodpile swung half door kitchen saw limpsy we...
940	met the little jackal coming gaily down the ro...	Bryant	met little jackal coming gaily road toward oh ...
99	one silly little hope is hopeless yet in what ...	Chesterton	one silly little hope hopeless yet hope asked ...
952	of nature there sprang from the sons of odin a...	Bryant	nature sprang sons odin race men became mighty...
704	armies bright which but th omnipotent none cou...	Milton	armies bright th omnipotent none could foiled ...

Figure 4: DataFrame of data set after stopwords removal.

1.2.3. Lemmatization and Stemming

Next, we removed all the similar root words using lemmatization function from the nltk library to avoid having multiple features for the same root word (e.g.: cats and cat). The data frame of data set after lemmatization is shown in figure 5.

	books_text_data	authors_labels	text_no_stopwords	lemmatized_text
694	fire then who created thee lamenting learn whe...	Milton	fire created thee lamenting learn uncreate the...	fire then who created thee lamenting learn whe...
964	close down in the cover of the others would ca...	Bryant	close cover others would call oh please dear s...	close down in the cover of the others would ca...
353	eagerly wrote down all three dates on their sl...	Carroll	eagerly wrote three dates slates added reduced...	eagerly wrote down all three date on their sla...
827	old man a cow and i can run away from you i ca...	Bryant	old man cow run away horse chased looked shoul...	old man a cow and i can run away from you i ca...
594	chants for thee the eternal march of thee a wa...	Whitman	chants thee eternal march thee war soldiers al...	chant for thee the eternal march of thee a war...

Figure 5: DataFrame of data set after lemmatization.

We also further created a column containing data after lemmatization and stop words removal, figure 6. Also, we tried to remove all the suffixes from the tokenized words using the stemming function from the nltk library as shown in figure 7.

	books_text_data	authors_labels	text_no_stopwords	lemmatized_text	lemm_nostopwords_text
320	five is twelve and four times six is thirteen ...	Carroll	five twelve four times six thirteen four times...	five is twelve and four time six is thirteen a...	five twelve four time six thirteen four time s...
87	you will destroy mankind you will destroy the ...	Chesterton	destroy mankind destroy world let suffice yet ...	you will destroy mankind you will destroy the ...	destroy mankind destroy world let suffice yet ...
192	face projected suddenly broad and brutal the c...	Chesterton	face projected suddenly broad brutal chin carr...	face projected suddenly broad and brutal the c...	face projected suddenly broad brutal chin carr...
692	let the earth put forth the verdant grass herb...	Milton	let earth put forth verdant grass herb yieldin...	let the earth put forth the verdant grass herb...	let earth put forth verdant grass herb yieldin...
477	crises labors beyond all others around me i he...	Whitman	crises labors beyond others around hear eclat ...	crisis labor beyond all others around me i hea...	crisis labor beyond others around hear eclat w...

Figure 6: DataFrame of data set after lemmatization and stopword removal.

	books_text_data	authors_labels	text_no_stopwords	lemmatized_text	lemm_nostopwords_text	stem_text
905	god sometimes the singers were divided into tw...	Bryant	god sometimes singers divided two great chorus...	god sometimes the singer were divided into two...	god sometimes singer divided two great chorus ...	god sometim the singer were divid into two gre...
864	park and look at the lovely flower beds they s...	Bryant	park look lovely flower beds seemed always fin...	park and look at the lovely flower bed they se...	park look lovely flower bed seemed always fini...	park and look at the love flower bed they seem...
610	fraud in the serpent speaking as he spake no g...	Milton	fraud serpent speaking spake ground enmity us ...	fraud in the serpent speaking a he spake no gr...	fraud serpent speaking spake ground enmity u k...	fraud in the serpent speak as he spake no grou...
126	be hansom cabs again if we are to keep him in ...	Chesterton	hansom cabs keep sight raced along gate elepha...	be hansom cab again if we are to keep him in s...	hansom cab keep sight raced along gate elephan...	be hansom cab again if we are to keep him in s...
917	news and all the village came to help they sta...	Bryant	news village came help started fires dug trenc...	news and all the village came to help they sta...	news village came help started fire dug trench...	news and all the villag came to help they star...

Figure 7: DataFrame of data set after stemming.

1.3. Text Analysis and Exploration

1.3.1. Chunks Count

As we chose 200 chunks from each book to create a data set, so each class will have the same number of chunks. This is confirmed by plotting the number of chunks per class as shown in figure 8.

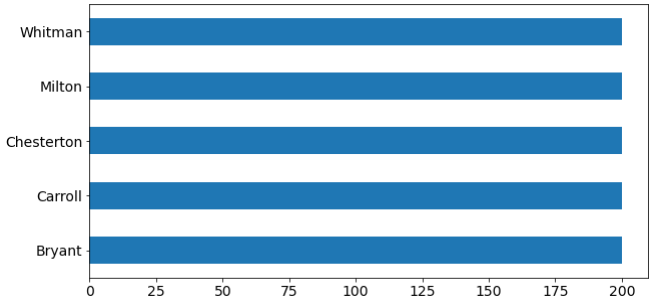

Figure 8: The number of chunks per class.

1.3.2. Length Analysis

Also, the length analysis has been performed as shown in figure 9. This confirmed that we have 100 words in each chunk in addition to other lengths shown in figure 9.

	books_text_data	authors_labels	labels_encoder	word_count	char_count	sentence_count	avg_word_length	avg_sentence_lenght
862	you always speak to me if everything is all ri...	Bryant	0	100	390	1	3.90	100.0
444	i bring thee muse to day and here all occupati...	Whitman	4	100	510	1	5.10	100.0
511	soul loves here the flowing trains here the cr...	Whitman	4	100	474	1	4.74	100.0
694	fire then who created thee lamenting learn whe...	Milton	3	100	496	1	4.96	100.0
798	none are to behold the judgement but the judge...	Milton	3	100	464	1	4.64	100.0

Figure 9: Length analysis.

The character count and distributions are shown in figure 10.

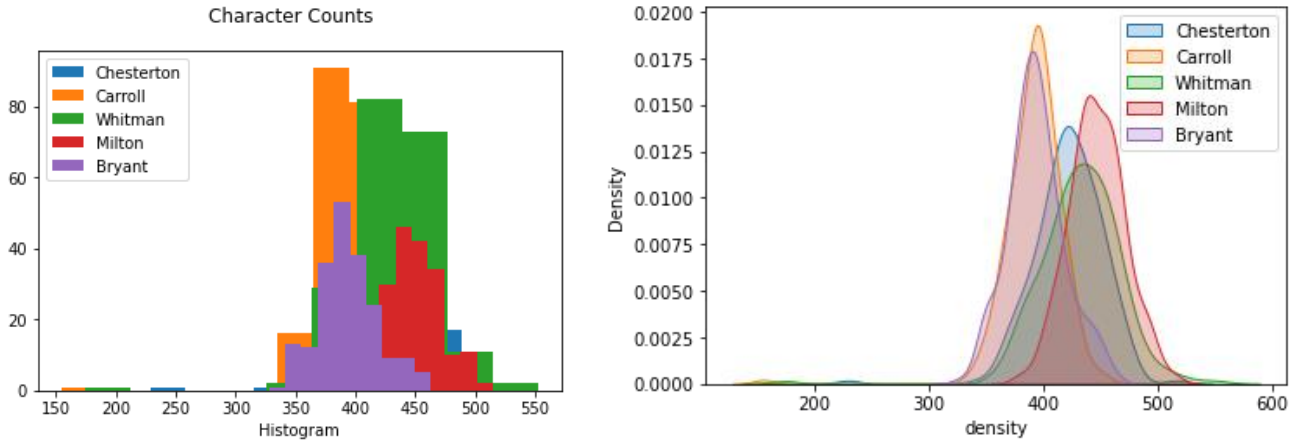

Figure 10: Character count and density.

1.3.3. Word Frequency

The word frequency has been plotted for most 10 frequent words using unigram and bigram as shown in figure 11 and figure 12. Visualization of word count is shown in figure 13 too.

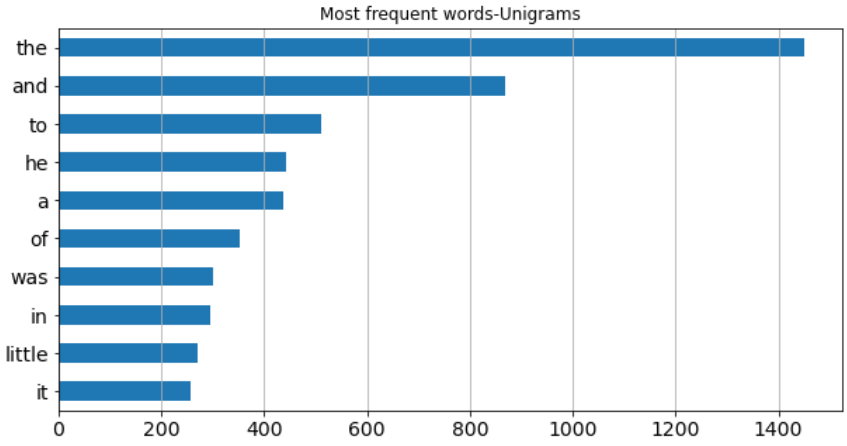

Figure 11: Most frequent words using unigram.

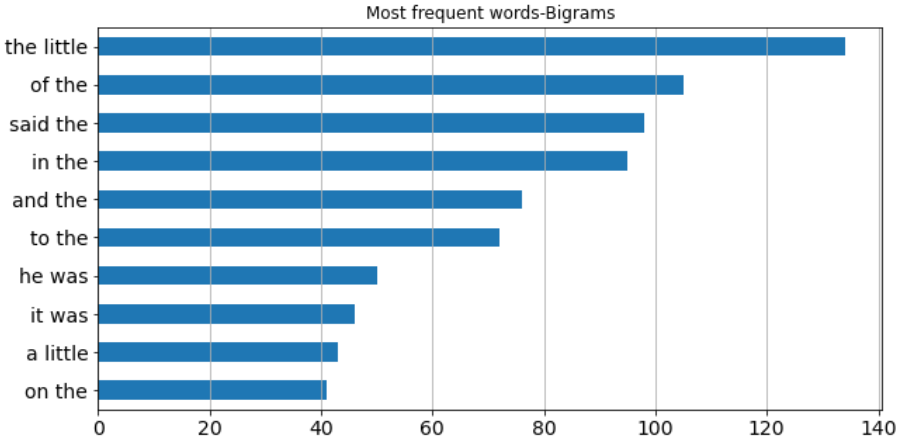

Figure 12: Most frequent words using bigram.

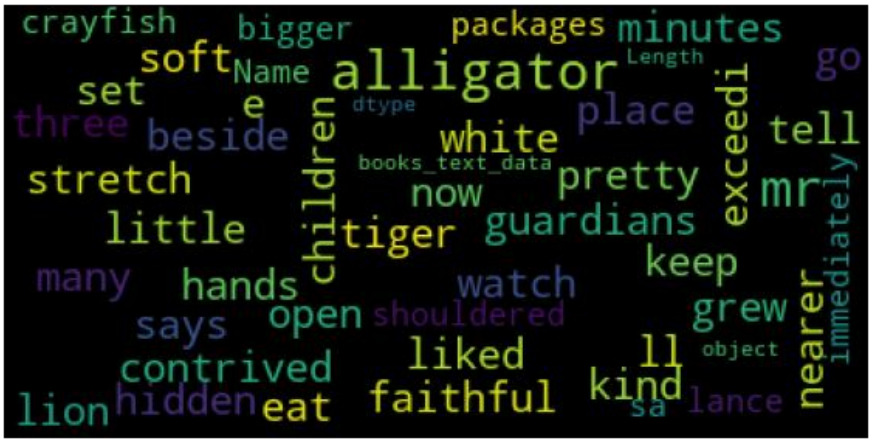

Figure 13: Visualization of word cloud.

2. Feature Engineering

In feature engineering, we introduced label encoding, dataset splitting for training, testing, and validation. oAlso, the transformation models such as bag of words and TF-iDF, n-gram have been introduced too.

2.1. Label Encoding

The authors labelled have been encoded to be transformed to a numerical value between 0 and n_{label}-1. Figure 14 shows the encoded labels in the data frame.

	books_text_data	authors_labels	labels_encoder	text_no_stopwords	lemmatized_text	lemm_nostopwords_text	stem_text
438	them and all thou that in all and over all and...	Whitman	4	thou incessant thou thou vital universal giant...	them and all thou that in all and over all and...	thou incessant thou thou vital universal giant...	them and all thou that in all and over all and...
451	give them their way and your songs outlaw d of...	Whitman	4	give way songs outlaw offenders scan kindred e...	give them their way and your song outlaw d off...	give way song outlaw offender scan kindred eye...	give them their way and your song outlaw d off...
471	poems being merely pictures the poems of the p...	Whitman	4	poems merely pictures poems privacy night men ...	poem being merely picture the poem of the priv...	poem merely picture poem privacy night men lik...	poem be mere pictur the poem of the privaci of...
648	my glory i behold in full resplendence heir of...	Milton	3	glory behold full resplendence heir might near...	my glory i behold in full resplendence heir of...	glory behold full resplendence heir might near...	my glori i behold in full resplend heir of all...
830	famous picture called the sower perhaps you ha...	Bryant	0	famous picture called sower perhaps seen putti...	famous picture called the sower perhaps you ha...	famous picture called sower perhaps seen putti...	famou pictur call the sower perhap you have se...

Figure 14: Labels encoding.

2.2. Splitting Data Set

The data set has been divided to training set and testing set as 85/15 as shown in figure 15.

```
# divide the data to training set and testing set 85/15
from sklearn.model_selection import train_test_split
df_books_train, df_books_test, df_books_labels_train, df_books_labels_test = train_test_split(
df_books['lemm_nostopwords_text'],df_books['labels_encoder'], test_size=0.15, random_state=0)
```

Figure 15: Splitting data set.

2.3. Bag of Words (BOW) Transformation

Bag of Words is a natural language processing text modelling technique. We cannot feed our text directly into algorithms, so we convert our text into numbers. This technique is used to process the text data into the bag of words which keeps a count on the total numbers of words used in the entire text. We can visualize the data by displaying the number of words counts mapping against the words.
For BOW feature extraction we made use of the function available in the ScikitLearn Library. In BOW, the countVectorization converts the number of repeated words into sparse matrix which can used further for modelling the data.

Advantages:

- The main advantage of bag of words is that it is very easy to implement and understand.
- This technique has a lot of flexibility for customization on the dataset.
- It is used on prediction problems like documentation classification and language modelling.

Disadvantage:

- If the same word is seen in upper case and lower case, BOW generates different tokens for them.
- Bag of words assumes that all words are independent of each other.
- BOW has low accuracy for sentiment analysis due to ignorance of grammar and semantics of the words.

2.4. TF-iDF Transformation.

TF-iDF stands for term frequency-inverse document frequency. It is a statistical technique which reflects how important a word is in a document. For TF-iTF, TfidfVectorizer() function was used

from the Sci-kit learn library. TF-iDF basically sums up the number of word occurrence in the document with the number of times it appears in the document.

Advantages:

- TF-iDF techniques result in a document that extracts the most descriptive words present inside a document.
- It can be used to measure the unique and relevant data in the document.
- It can be calculated easily.

Disadvantages:

- This technique is slow for large data as it computes the document similarity directly in the word-count space.
- Semantic similarities between words are not used at all.
- It assumes that the counts of each word provide an independent verification of resemblance.

A quick comparison between TF-iDF and BOW is show below in the figure 16.

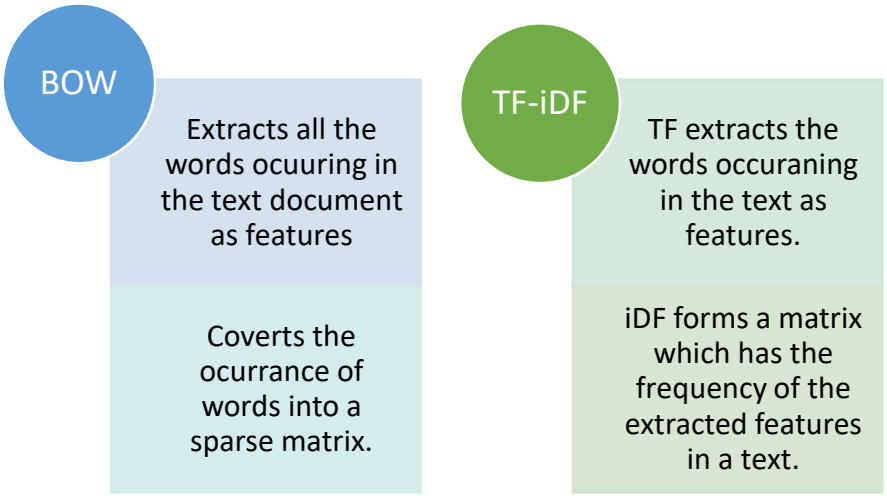

Figure 16: Labels encoding.

3. Machine Learning Models

In this project, we implemented three machine learning classification models and compare them to find the best one. The three models are listed below:

- Support Vector Machine
- K Nearest Neighbors
- Decision Tree

The algorithms are implemented using Scikit learn library functions. The methodology used to train and test each model is as follows:

- At first, check what parameters the model has.
- Implement the baseline model and measure its accuracy.
- Perform 10-fold cross validation and check the accuracy.
- Perform grid search cross validation process to find if we can improve the model performance by tuning some parameters.
- When getting the best parameters, the confusion matrix, classification report, accuracy, bias, variance, and MSE are used to analyze the model performance on both training and test data.
- Compare the accuracy of baseline model and tuned model to check if we got better performance.

- The best model will be chosen based on testing accuracy, bias, and variance.

3.1. Support Vector Machine (SVM) Model

This model tries to find the most optimal hyperplane with the utmost margin from each Support Vector (The borderline data points from each class). Hyperplanes act like decision extremity points between different classes.

3.1.1. SVM Cross-Validation and Grid Search

This model has been evaluated by 10-fold cross validation; the average accuracy was 90 %. That accuracy is high, but let's see if better parameters can be found using grid search.

For grid search cross validation, a parameters grid was created including:

- Penalty parameter (C).
- Kernel coefficient (gamma).
- Degree of the polynomial.
- Kernel type.
- Probability

After performing the grid search, the best obtained parameters SVM model are {C: 1, kernel: linear, probability: True}.

3.1.2. SVM Model Results and Error Analysis

The confusion matrix, classification report, accuracy, bias, variance, and MSE are used to evaluate the model performance on both training and test data. This model trained and tested using BOW and TF-iDF. The metrics summery are shown in table 3. The MSE, Bias, and Var were not calculated for SVM with BOW due to the long runtime and for the reason that BOW did not provide a great result compared with TF-iDF. When using BOW, the SVM model tends to be overfit since the training accuracy is 100 % and its testing accuracy is lower. The SVM model has testing accuracy of 90 and precision, recall and fscore of 89 %. It recoded low bias and variance as shown in the summary table 3. If we increase the features number and data set size and try playing with transformation techniques, we can enhance the accuracy and further decrease the bias.

Table 3: Metrics summery of SVM model

SVM								
Feature	Accuracy		Precision	Recall	F1-score	MSE	Bias	Var
	Training	Testing						
BOW	1.00	0.90	0.9	0.9	0.9	-	-	-
TF-iDF	0.96	0.90	0.89	0.89	0.89	0.487	0.296	0.19

The confusing matrix is shown in figure 17. As noticed, we have 150 samples for testing, and they are not equally distributed over all classes since the second class (Carroll) has 40 samples with 37 samples predicted correctly. This unequal distribution is due to random splits. In general, the matrix shows the SVM has low mis-predicted percentages over all classes.

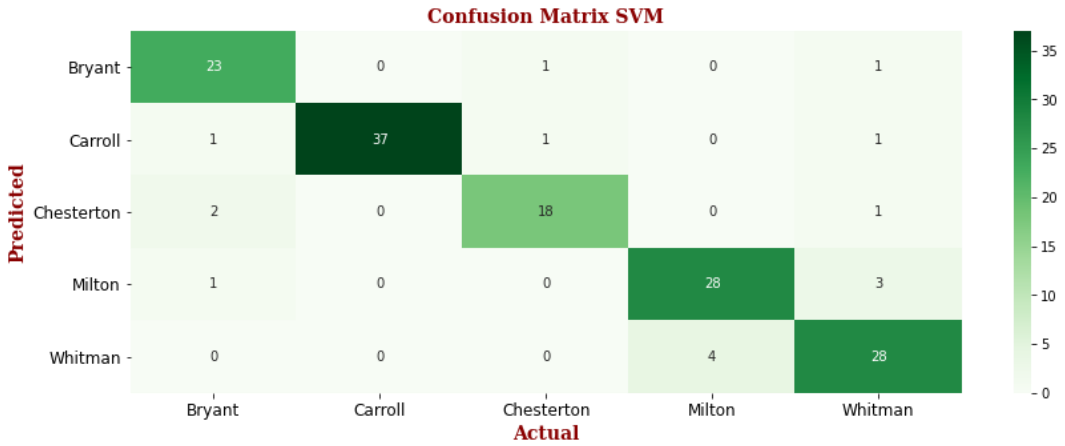

Figure 17: Confusing matrix for SVM.

3.1.3. Advantages and disadvantages of SVM

The advantages and disadvantages of SVM model are summarized in figure 18.

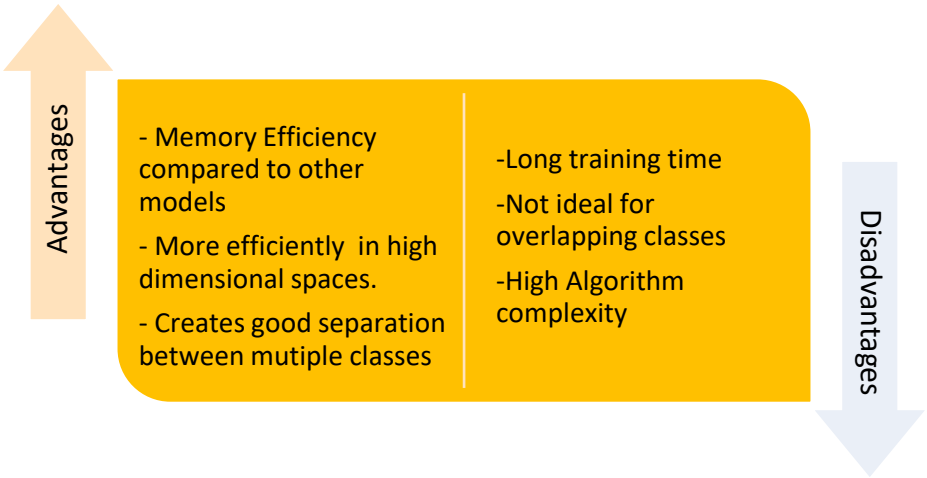

Figure 18: Advantages and disadvantages of SVM.

3.2. Decision Tree (DT)

Decision Tree is one of the most popular machine learning algorithms for supervised learning. This algorithm is used to solve both regression and classification problems. Decision trees have a tree structure that has leaves which represents class labels and branches which represents the co-occurrence of features that lead to those class labels.

3.2.1. DT Cross-Validation and Grid Search

This model has been evaluated by 10-fold cross validation; the average accuracy was 72 %. That accuracy is not too high, so let's see if better parameters can be found using grid search. For grid search cross validation, a parameters grid was created as show in table 4 below:

Table 4: Grid search parameters to tune DT model.

criterion	min_samples_split	max_depth:	min_samples_leaf	max_leaf_nodes
[gini, entropy]	[2, 10, 20]	[None, 2, 5, 10]	[1, 5, 10]	[None, 5, 10, 20]

After performing the grid search, the best obtained parameters SVM model are listed in table 4 below.

Table 5: The best obtained parameters SVM model.

criterion	min_samples_split	max_depth:	min_samples_leaf	max_leaf_nodes
gini	2	None	1	20

3.2.2. DT Model Results and Error Analysis

The confusion matrix, classification report, accuracy, bias, variance, and MSE are used to evaluate the model performance on both training and test data. This model trained and tested using BOW and TF-iDF. The metrics summery are shown in table 6. The MSE, Bias, and Var were not calculated for SVM with BOW due to the long runtime and for the reason that BOW did not provide a great result compared with TF-iDF. The DT model has testing accuracy of 75, precision of 79 %, recall and fscore of 75 %. It recoded high bias and variance as shown in the summary table 6. This is due to the model capacity and the number of features.

Table 6: Metrics summery of DT model.

DT								
Feature	Accuracy		Precision	Recall	F1-score	MSE	Bias	Var
	Training	Testing						
BOW	0.81	0.72	0.78	0.71	0.72	-	-	-
TF-iDF	0.81	0.75	0.79	0.75	0.75	1.52	0.92	0.6

The confusing matrix is shown in figure 19. Also, As noticed, unequal distribution is shown in the matrix as mentioned previously. In general, the matrix shows the SVM has higher mis-predicted percentages over all classes.

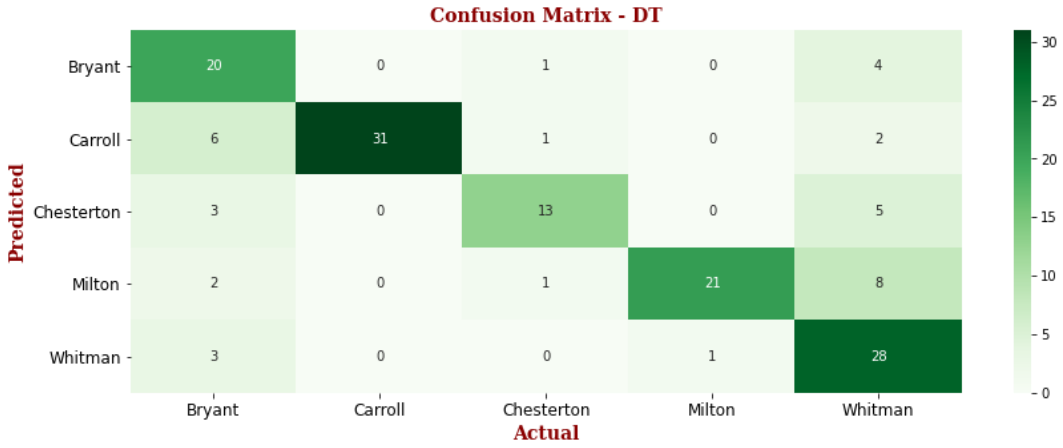

Figure 19: Confusing matrix for DT.

3.2.3. Advantages and disadvantages of DT

The advantages and disadvantages of DT model are summarized in figure 20.

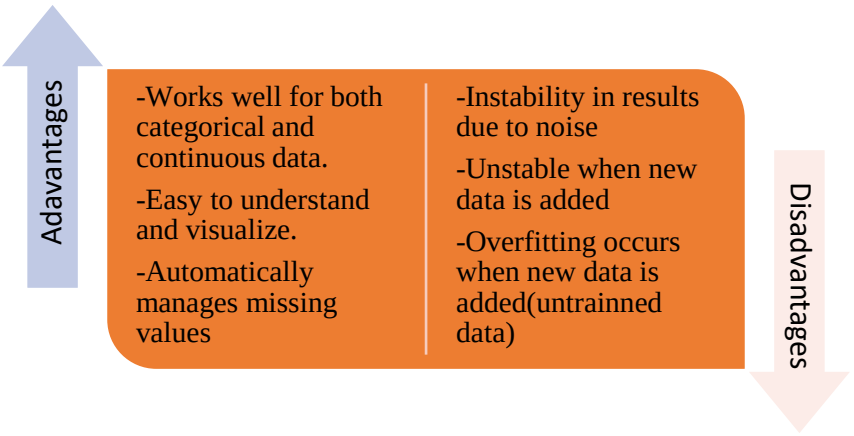

Figure 20: Advantages and disadvantages of DT.

3.3. K-Nearest Neighbors (KNN)

This algorithm is based on supervised learning techniques. It is mostly used to solve classification problems and some regression problems as well.

3.3.1. KNN Cross-Validation and Grid Search

This model has been evaluated by 10-fold cross validation; the average accuracy was 83 %. That accuracy is not too high, so let's see if better parameters can be found using grid search. The grid search cross validation was used to check if we can tune the number of neighbors (k). After performing the grid search, the best obtained parameter k is 93.

3.3.2. KNN Model Results and Error Analysis

The confusion matrix, classification report, accuracy, bias, variance, and MSE are used to evaluate the model performance on both training and test data. This model trained and tested using BOW and TF-iDF. The metrics summery are shown in table 7. The MSE, Bias, and Var were not calculated for SVM with BOW due to the long runtime and for the reason that BOW did not provide a great result compared with TF-iDF. This model has very low performance when using BOW. When using TF-iDF, the KNN records 87 % testing accuracy and around 86 % for precision, recall, and f1-score. It has relatively higher bias and variance.

Table 7: Metrics summery of KNN model.

KNN								
Feature	Accuracy		Precision	Recall	F1-score	MSE	Bias	Var
	Training	Testing						
BOW	0.4	0.4	0.6	0.4	0.34	-	-	-
TF-iDF	0.86	0.87	0.86	0.86	0.86	0.66	0.43	0.23

The confusing matrix is shown in figure 21.

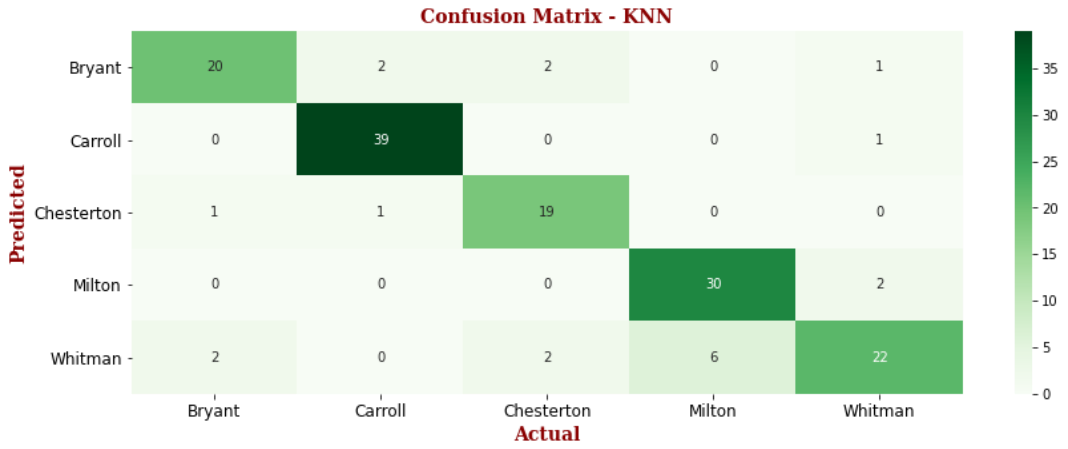

Figure 21: Confusing matrix for KNN.

3.3.3. Advantages and disadvantages of KNN

The advantages and disadvantages of KNN model are summarized in figure 20.

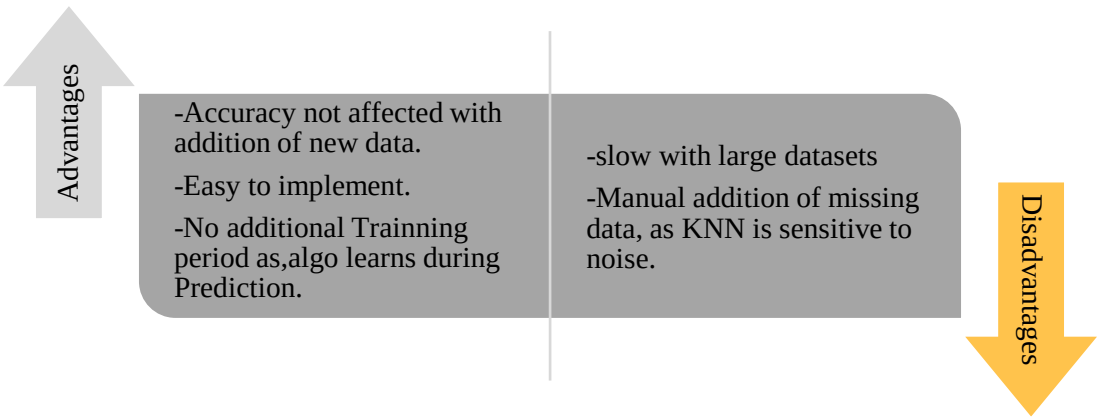

Figure 22: Advantages and disadvantages of KNN.

4. Best Model Selection

4.1. Models Summary and Comparison

The table 8 summarizes the metrics of all three models using TF-iDF. As shown, the SVM has the highest metrics and lowest bias and variance. DT has the lowest metrics and highest bias and variance. The KNN is relatively close to SVM, but it has more bias and variance. Therefore, the SVM is chosen to be the champion model.

There are some parameters that affect the model's performance such as low training features and the size of each chunks is small. Also, the chunks are formed by partitioning 100 words and this may not enough. Also, the partitioning is not efficient to capture the full sentences, it is only based on 100 words regardless of if the sentence is complete or not. Also, after lemmatization and transformation, the sentences may lose some features. So, the performance of the model and its metrics depend on many things such as:

- The features number.
- Samples size and length.
- The model capacity.
- The transformation technique (TF-iDF or BOW).
- The size of training and testing splits.
- The distribution of samples over classes.

Table 8: Metrics summery of all models using TF-iDF.

TF-iDF								
Feature	Accuracy		Precision	Recall	F1-score	MSE	Bias	Var
	Training	Testing						
SVM	0.96	0.9	0.89	0.89	0.89	0.48	0.3	0.18
KNN	0.86	0.87	0.86	0.86	0.86	0.66	0.43	0.23
DT	0.80	0.75	0.79	0.75	0.75	1.52	0.92	0.6

When using BOW, the SVM still has the highest testing accuracy, but is seems overfit. The KNN recorded the lowest metrics compared with others and compared with its metrics with TF-iDF. The DT almost recorded the same metrics as in TF-iDf. The TF-iDF is chosen to be the best over BOW due to its better performance.

Table 9: Metrics summery of all models using BOW.

BOW					
Model	Accuracy		Precision	Recall	F1-score
	Training	Testing			
SVM	1.00	0.90	0.90	0.90	0.90
KNN	0.4	0.43	0.60	0.41	0.34
DT	0.81	0.72	0.78	0.71	0.72

4.2. Best Model Interpretation

At this point we have selected the SVM as our preferred model to do the predictions. We will now study its behaviour by analysing misclassified text. Hopefully this will give us some insights on the way the model is working.

4.2.1. Creating Pipeline

A pipeline has been created for the champion model as shown below. We trained this model to training data.

```
# Create a pipeline
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import TfidfTransformer

SVM_Pipeline = Pipeline([
    ('vect', CountVectorizer()),
    ('tfidf', TfidfTransformer()),
    ('clf', best_SVM)])

#Training
SVM_Pipeline.fit(df_books_train, df_books_labels_train)
```

4.2.2. Scoring Pipeline

The created pipeline has been scored. In the code below, we can ask the machine to predict the author of a given text, and it will return the authour name.

```
# Scoring the pipeline
docs_new = ['in a small tug up the silent thames armed with',
            'watch like faithful guardians beside the open',
            'can tell you and other packages bigger that sa']
predicted = SVM_Pipeline.predict(docs_new)
for doc, category in zip(docs_new, predicted):
    print('The author of following text ({}) is {}'.format(doc,''.join(le.inverse_transform([category]))))
```

```
The author of following text (in a small tug up the silent thames armed with) is Chesterton
The author of following text (watch like faithful guardians beside the open) is Bryant
The author of following text (can tell you and other packages bigger that sa) is Whitman
```

Now, let us check if the model can predict the authors of testing data. In figure 23 below, a random sample of predicted authors have been shown.

	books_text_data	authors_labels	Predicted_Authors
408	spin around turning always towards mecca i see...	Whitman	Whitman
412	earth does this is curious and may not be real...	Whitman	Whitman
958	clown answered very sadly that it was just as ...	Bryant	Bryant
531	city but these and all and the brown and sprea...	Whitman	Whitman
319	ve got to grow up again let me see how is it t...	Carroll	Carroll
939	lazy man what became of him and his crab no on...	Bryant	Bryant
636	so might the wrath fond wish couldst thou supp...	Milton	Milton
993	found her sitting in a tent by the side of an ...	Bryant	Bryant
859	come along home like this all right mammy said...	Bryant	Bryant
495	all all is for individuals all is for you no c...	Whitman	Whitman

Figure 23: Random sample of predicted authors.

The misclassified text partitions are shown in figure 24. The number of misclassified partitions is 16. They are around 10.67 % of the total predicted partitions 150 partitions.

	books_text_data	authors_labels	Predicted_Authors
31	car and after staring wildly at the wintry sky...	Chesterton	Bryant
310	she thought it must be the right house because...	Carroll	Chesterton
883	one second after midnight it will be too late ...	Bryant	Chesterton
45	the great poisoning princes of the renaissance...	Chesterton	Bryant
698	foretold what shall befall him or his children...	Milton	Whitman
783	spangling the hemisphere then first adorned wi...	Milton	Bryant
141	heard in silence the hum of insects and the di...	Chesterton	Whitman
982	the sea king was hard of heart like frode and ...	Bryant	Whitman
501	today thou art all over set in births and joys...	Whitman	Milton
785	reins from the equinoctial road like distant b...	Milton	Whitman
317	stand on your head do you think at your age it...	Carroll	Bryant
466	glistening and rolling tufts of straw sands fr...	Whitman	Milton
545	its rules precise and delicatesse the lyrist s...	Whitman	Milton
630	him a cumbrous train of herds and flocks and n...	Milton	Whitman
520	you copings and iron guards you windows whose ...	Whitman	Milton
278	the dream of wonderland of long ago and how sh...	Carroll	Whitman

Figure 24: misclassified text partitions.

Now, we can predict the author of some random partitions belong to our data set, and then let the model predict a set of text that not belong to our data set but taken from other books for the same authors. A prediction function has been defined as shown below.

```
# define a prediction function
def prediction(text):
    predicted = SVM_Pipeline.predict(text)
    for doc, category in zip(text, predicted):
        print('The author of predicted text: {}'.format(''.join(le.inverse_transform([category]))))
    pred_proba = SVM_Pipeline.predict_proba(text)
    for i in range(len(text)):
        print("The corresponding conditional probability :{:.2f}".format(pred_proba[i].max()*100))
```

Then, we predicted the author of some random partitions that belong to our data set and calculated the conditional probability; the result is shown below.

```
text=[df_books['books_text_data'].iloc[300],df_books['books_text_data'].iloc[350],
      df_books['books_text_data'].iloc[400],
     df_books['books_text_data'].iloc[440],df_books['books_text_data'].iloc[600]]
prediction(text)
```

```
The author of predicted text: Carroll
The author of predicted text: Carroll
The author of predicted text: Whitman
The author of predicted text: Whitman
The author of predicted text: Milton
The corresponding conditional probability :99.99
The corresponding conditional probability :99.97
The corresponding conditional probability :99.88
The corresponding conditional probability :99.94
The corresponding conditional probability :100.00
```

Then, we can check the authors by comparing the result by the actual authors. As shown prediction gave us the same actual authors.

```
# check the authors by comparing the result by the actual authors
text_authors=[df_books['authors_labels'].iloc[300],
              df_books['authors_labels'].iloc[350],
              df_books['authors_labels'].iloc[400],
              df_books['authors_labels'].iloc[440],
              df_books['authors_labels'].iloc[600]]
print(text_authors)
# we prediction gave us the same actual authors
```

```
['Carroll', 'Carroll', 'Whitman', 'Whitman', 'Milton']
```

Now, we can predict text that not related to our data set, but it is taken from books for the same our authors. The code piece below shows the process of taking some paragraphs from other books.

```
b1 = nltk.corpus.gutenberg.raw('austen-emma.txt')
b2 = nltk.corpus.gutenberg.raw('chesterton-brown.txt')
b3 = nltk.corpus.gutenberg.raw('blake-poems.txt')
# Create a list of books
books=[b1,b2,b3]
new_authors=['Austen','Chesterton','Blake']
new_books=['austen-emma','chesterton-brown','blake-poems']
austen_emma='''The want of Miss Taylor would be felt every hour of every day.\nShe recalled
her past kindness--the kindness, the affection of sixteen\nyears--how she had taught and how
she had played with her from five\nyears old--how she had devoted all her powers to attach and
amuse\nher in health--and how nursed her through the variousillnesses\nof childhood'''
chesterton_brown='''A tantalus\ncontaining three kinds of spirit, all of a liqueur excellence,
\nstood always on this table of luxury; but the fanciful have asserted\nthat the whisky, brandy,
and rum seemed always to stand at the same level.\nPoetry was there:  the left-hand corner of the
room was lined with\nas complete a set of English classics as the right hand could show\nof English
and foreign physiologists'''
blake_poems='''The little boy weeping sought.\n \n \n LAUGHING SONG\n \n When the green woods laugh
with the voice of joy,\n And the dimpling stream runs laughing by;\n When the air does laugh with our merry
wit,\n And the green hill laughs with the noise of it;\n \n when the meadows laugh with lively green,\n And the
grasshopper laughs in the merry scene,\n When Mary and Susan and Emily\n With their sweet round mouths sing "Ha, ha he'''
```

After cleaning the text and predicting it, we got the following results:

```
austen_emma=re.sub('[^a-zA-Z]',' ', austen_emma)
chesterton_brown=re.sub('[^a-zA-Z]',' ', chesterton_brown)
blake_poem=re.sub('[^a-zA-Z]',' ', blake_poems)
docs_new=[austen_emma,chesterton_brown,blake_poem]
```

```
prediction(docs_new)
```

```
The author of predicted text: Whitman
The author of predicted text: Chesterton
The author of predicted text: Whitman
The corresponding conditional probability :78.27
The corresponding conditional probability :78.02
The corresponding conditional probability :73.73
```

As seen, the model only classified the second text to its author (Chesteron) and classified other authors wrongly and the probability is reduced too. We need to reduce the conditional probability belonging to every class because our text data set consists of 5 different books. Therefore, if we passed a text paragraphs that do not belong to one of these books, we will get a wrong prediction.

In addition, we tried to play with features and samples size to bring the accuracy of champion model to around 20 %, but we could not reach to that due to the long runtime each time we modified the features and samples numbers and SVM has slow training process. The lowest accuracy we got so far is around 45 % when reducing the number of features and samples, the corresponding bias and variance were increased consequently. Also, playing with pre-processing text resulted with overfitting. Therefore, to enhance the model performance, we need to increase the number of features and increase the data set size and make a threshold for that to keep an excellent performance for the model.

5. PCA and t-SNE Decomposition Plots

Figure 25 shows the PCA and t-SNE decomposition plots. From the graph we can see the Carroll and Milton components hold some information more than other classes in PCA plot. The t-SNE plot gives more distribution.

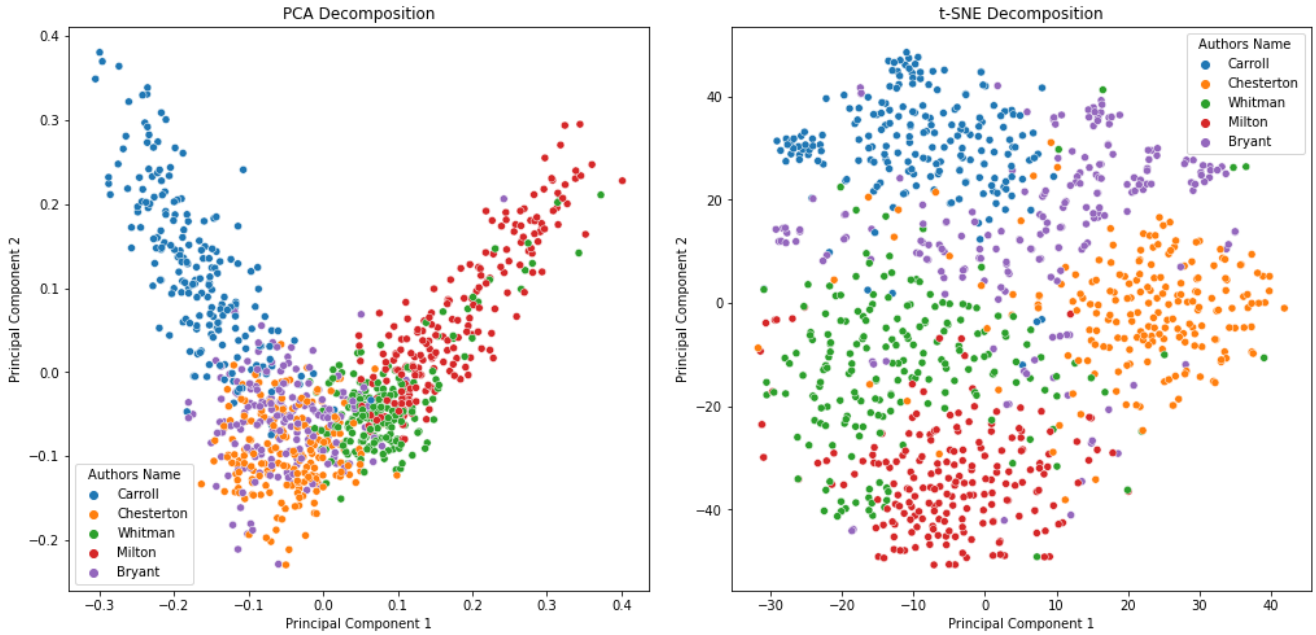

Figure 25: PCA and t-SNE decomposition plots.

6. Programs README

The prerequisites to follow the code of this assignment are python version 3.7 and jupyter notebook. For this assignment, scikit-learn, re, nltk, pandas, numpy, seaborn, and mlxtend are mainly used. The latest version of mlxtend should be downloaded in order to work properly. The code was written serially in one draft. Thus, each step depends on the step before it. So, when running a specific piece of code, you should run the pre-step and so on.

7. Conclusion

In this assignment, we have learned the text classification. We learned about important concepts like bag of words, TF-IDF and three algorithms SVM, KNN and DT. For our data set, the SVM is chosen to be the champion model due to its high metrics and low bias and variance values. The models have been evaluated by performing ten-fold cross-validation. We also performed grid search for parameters tuning in order to get better performance. We tried to play with features and samples size to bring the accuracy of champion model to around 20 %, but we could not reach to that due to the long runtime each time we modified the features and samples numbers and SVM has slow training process. The lowest accuracy we got is around 45 % when reducing the number of features and samples, the corresponding bias and variance were increased consequently. Therefore, to enhance the model performance, we need to increase the number of features and increase the data set size and make a threshold for that to keep an excellent performance for the model.

We gauged the performance of models by the common classification metrics such as the confusion matrix, classification report, accuracy, bias, variance, and MSE for both training and testing data.

There are some parameters that affect the model's performance such as:

- The features number.
- Samples size and length.

- The model capacity.
- The transformation technique (e.g., TF-iDF, BOW).
- The size of training and testing splits.
- The distribution of samples over classes.

Finally, we came up with some insights that may help us to improve the performance such as:

- Play around with the data pre-processing steps.
- Try other Word Vectorization techniques or combine more than one.
- Enhance the partition process.
- Increase the dataset size.
- Increase the length of each partition.
- Play with features number.
- Samples size and length
- Choose a model with excellent capacity.
- Ply with the size of training and testing splits.
- Pay more attention for the distribution of samples over classes.

8. Refences

[1] "Model Selection in Text Classification | by Christophe Pere | Towards Data Science." https://towardsdatascience.com/model-selection-in-text-classification-ac13eedf6146 (accessed Feb. 08, 2021).

[2] "Multi-Class Text Classification Model Comparison and Selection | by Susan Li | Towards Data Science." https://towardsdatascience.com/multi-class-text-classification-model-comparison-and-selection-5eb066197568 (accessed Feb. 08, 2021).

[3] "A guide to Text Classification(NLP) using SVM and Naive Bayes with Python | by Gunjit Bedi | Medium." https://medium.com/@bedigunjit/simple-guide-to-text-classification-nlp-using-svm-and-naive-bayes-with-python-421db3a72d34 (accessed Feb. 08, 2021).

[4] "A Comprehensive Guide to Understand and Implement Text Classification in Python." https://www.analyticsvidhya.com/blog/2018/04/a-comprehensive-guide-to-understand-and-implement-text-classification-in-python/ (accessed Feb. 08, 2021).

[5] "Text Classification with Python and Scikit-Learn." https://stackabuse.com/text-classification-with-python-and-scikit-learn/ (accessed Feb. 08, 2021).

[6] "Text Classification in Python. Learn to build a text classification… | by Miguel Fernández Zafra | Towards Data Science." https://towardsdatascience.com/text-classification-in-python-dd95d264c802 (accessed Feb. 08, 2021).