DTI/GNG5125 Data Science Applications
Classification Assignment (Group)

By Group 7:
Kesha Shah
Leenanci Parmar
Prashant Kaushik
Vikram Khanzode

Present to:
Dr. Arya Rahgozar

February 3rd, 2022
University of Ottawa
Faculty of Engineering

1.Introduction

Classification is a predictive modelling task in machine learning where a class label is predicted for a given example of input data. In this project, we are focusing on the three-classification algorithms named KNN, SVM, and Naïve Bayes, giving importance to the pre-processing and comparison between these three algorithms. After referring to different research papers based on classification algorithms and going through various methodologies, we found the SVM algorithm as the most suitable one for our project.

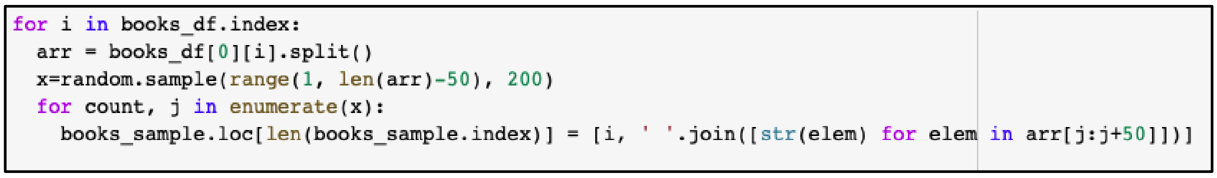
2.Dataset

We have taken five different samples of Gutenberg digital books, which are of five different authors. The names of the books from horror and one related to birds are as below:

1.The Strange Case Of Dr. Jekyll And Mr. Hyde, by Robert Louis Stevenson
2.Dracula by Bram Stoker
3.Metamorphosis by Franz Kafka
4.The Turn of the Screw by Henry James
5.The Bird Study Book by Thomas Pearson

3.Data preparation

With the use of the NLTK library, we read the books and then created random 200 samples of each book which contains 100 words. Each random sample is labelled by the author's name. Figure 1 illustrates the code to prepare the data.



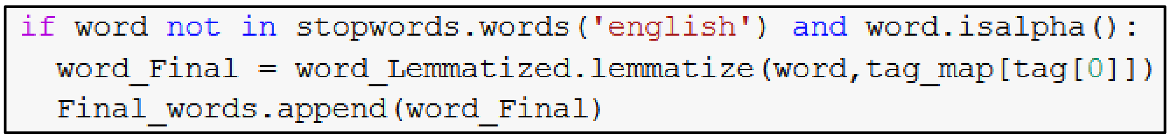
```
for i in books_df.index:
  arr = books_df[0][i].split()
  x=random.sample(range(1, len(arr)-50), 200)
  for count, j in enumerate(x):
    books_sample.loc[len(books_sample.index)] = [i, ' '.join([str(elem) for elem in arr[j:j+50]])]
```

Figure 1. Prepare the data

4.Pre-processing

Data preprocessing is the procedure for preparing raw data for use in a machine learning model. It's the first and most important stage in building a machine learning model. For this project, we have removed the stop words and garbage characters.

In addition we've performed lemmatization to convert words into their lemma form.Figure 2 shows the implementation to remove stop words and perform lemmatization.



```
if word not in stopwords.words('english') and word.isalpha():
  word_Final = word_Lemmatized.lemmatize(word,tag_map[tag[0]])
  Final_words.append(word_Final)
```

Figure 2. Remove stop words and perform Lemmatization

5.Feature engineering

The process of turning raw data into a format or structure that is more suitable for model building and data discovery, in general, is known as data transformation. It's a crucial phase in feature engineering that makes finding insights easier. For this project, we have transformed the pre-processed data into BOW and TF-IDF. BOW and TF-IDF are both methods for converting text phrases to numeric vectors.

5.1 BOW

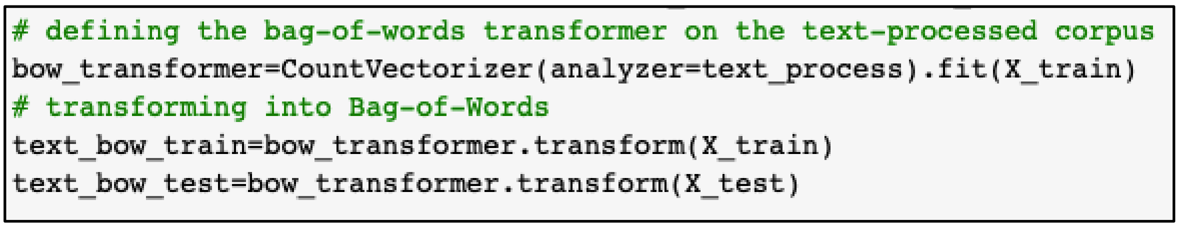
The Bag of Words (BoW) model is the most basic type of numerical text representation. A phrase can be represented as a bag of words vector, just like the term itself (a string of numbers).

For example, BOW for below two sentences is as follow:
Sentence 1: It is spooky and good
Sentence 2: It is not good and is slow
We will first build a vocabulary from all the unique words in the above two sentences. The vocabulary consists of these 7 words: ‘It’, ‘is’, ‘spooky’, ‘and’, ‘good’, ‘not’, ‘slow’.

We can now take each of these words and mark their occurrence in the two sentences above with 1s and 0s. This will give us 2 vectors for 2 sentences:

	1 It	2 is	3 spooky	4 and	5 good	6 not	7 slow	Length of sentence (in words)
Sentence 1	1	1	1	1	1	0	0	5
Sentence 2	1	2	0	1	1	1	1	7

Vector of Sentence 1: [1 1 1 1 1 0 0]
Vector of Sentence 2: [1 2 0 1 1 1 1]



```
# defining the bag-of-words transformer on the text-processed corpus
bow_transformer=CountVectorizer(analyzer=text_process).fit(X_train)
# transforming into Bag-of-Words
text_bow_train=bow_transformer.transform(X_train)
text_bow_test=bow_transformer.transform(X_test)
```

Figure 3. BOW

5.2 TF-IDF

A numerical metric called term frequency-inverse document frequency is meant to show how essential a word is to a document in a collection or corpus.

Term frequency-The frequency of a term t mentioned in document d is measured by TF:

$$tf_{t,d} = \frac{n_{t,d}}{Number\ of\ terms\ in\ the\ document}$$

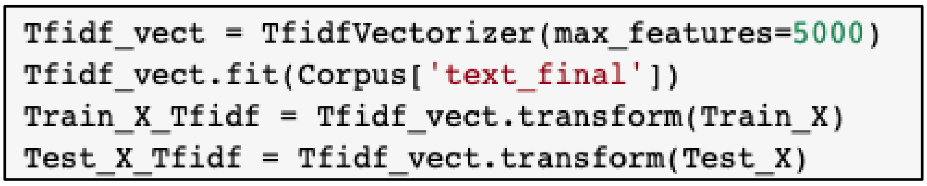
Here, in the numerator, n is the number of times the term “t” appears in the document “d”. Thus, each document and term has its own TF value.

Inverse Document Frequency-IDF is how important a term is. Computing TF is not sufficient to understand the significance of words. Therefore, IDF is also necessary.

$$idf_t = \log \frac{number\ of\ documents}{number\ of\ documents\ with\ term\ 't'}$$

TF-IDF is computed as below:

$$(tf\_idf)_{t,d} = tf_{t,d} * idf_t$$



```
Tfidf_vect = TfidfVectorizer(max_features=5000)
Tfidf_vect.fit(Corpus['text_final'])
Train_X_Tfidf = Tfidf_vect.transform(Train_X)
Test_X_Tfidf = Tfidf_vect.transform(Test_X)
```

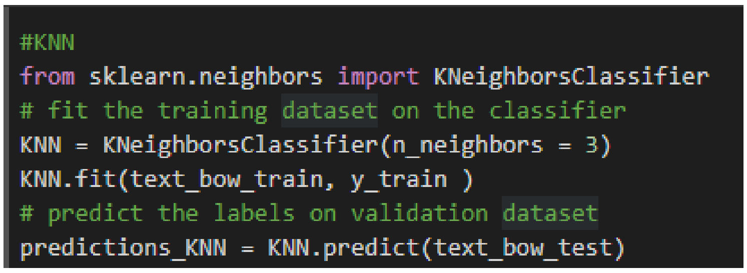
Figure 3. TF-IDF implementation

6.Classification models

6.1 KNN

The K-Nearest Neighbour algorithm is based on the Supervised Learning technique and is one of the most basic Machine Learning algorithms.

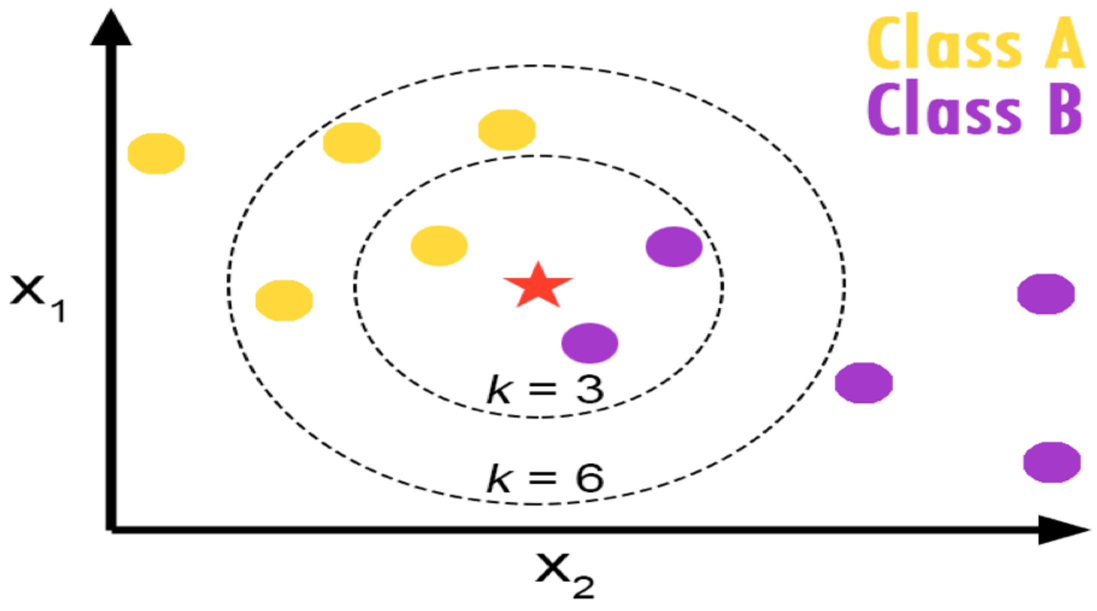
This method assumes that the new case/data and existing cases are similar and places the new case in the category that is most similar to the existing categories. It stores all available data and classifies a new data point based on its similarity to the existing data. This means that new data can be quickly sorted into a well-defined category using the K-NN method.



```
#KNN
from sklearn.neighbors import KNeighborsClassifier
# fit the training dataset on the classifier
KNN = KNeighborsClassifier(n_neighbors = 3)
KNN.fit(text_bow_train, y_train )
# predict the labels on validation dataset
predictions_KNN = KNN.predict(text_bow_test)
```

Figure 4. KNN implementation

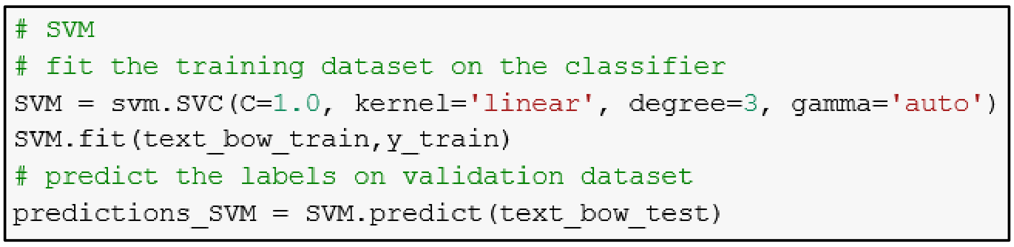
The K-NN approach can be used for both regression and classification, but it is more commonly utilised for classification tasks.



6.2 SVM

SVM is a supervised machine learning technique that is used for both classification and regression.The goal of the SVM algorithm is to find a hyperplane in an N-dimensional space that categorises data points clearly. The hyperplane's size is determined by the number of features.

Here, we use SVM to classify the authors and the features used are BOW vectors and TF-IDF vectors which are given to train the model. Figure 4. shows the code snippet for applying SVM.



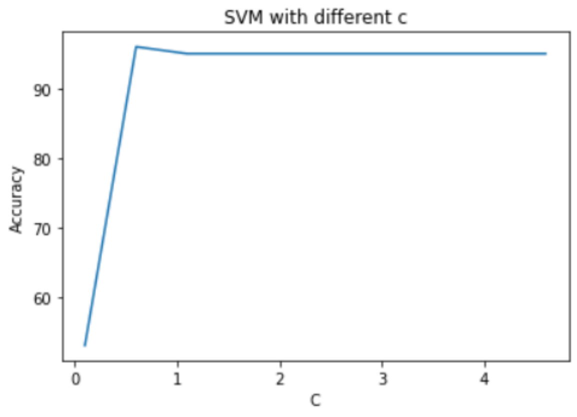
```
# SVM
# fit the training dataset on the classifier
SVM = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='auto')
SVM.fit(text_bow_train,y_train)
# predict the labels on validation dataset
predictions_SVM = SVM.predict(text_bow_test)
```

Figure 4. Applying SVM using scikit library in python

The following parameters[3] are used to fine tune the model and increase accuracy:

1) C: Regularization parameter. C increases the tendency of the model to overfit the data and as C decreases it is more prone to underfit the data. C is large then there is high variance and low bias, and if C is small then there is high bias and low variance.
2) kernel: Specifies the kernel type to be used in the algorithm. This impacts the loss function which is used while training.

In this assignment, mainly kernel and parameter C were varied to increase the accuracy. Figure 5. shows the line chart after varying c with different types of kernel on features.



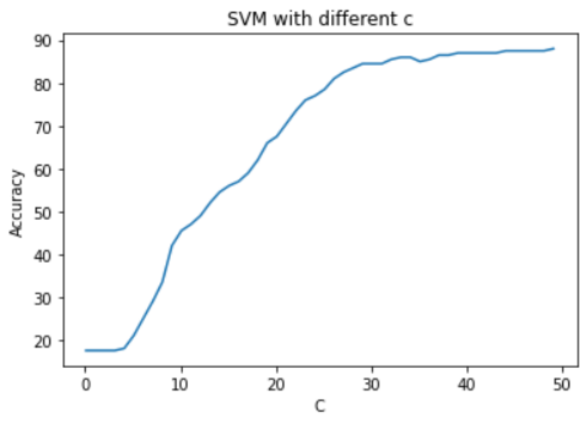


Figure 5a) kernel='linear' and TFIDF Figure 5b)kernel='sigmoid' and BOW

Highest accuracy using SVM for this classification problem was achieved by using linear kernel and c=0.6 for TF-IDF features.

6.3 Naive Bayes

The Bayes' Theorem is used to create a collection of classification algorithms known as Naive Bayes classifiers. A Naive Bayes classifier is a probabilistic machine learning model that's used for classification tasks. The crux of the classifier is based on the Bayes theorem.

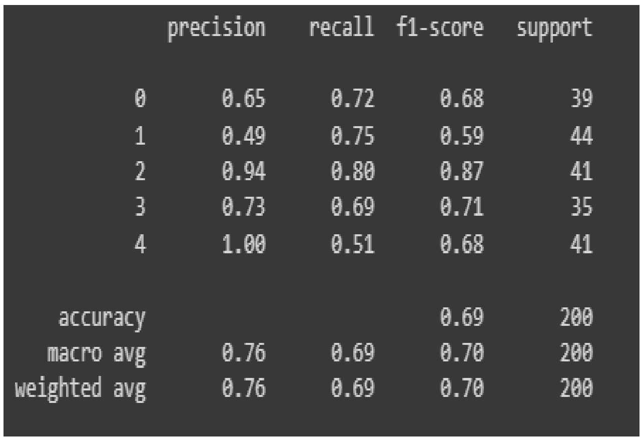
$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Using Bayes theorem, we can find the probability of A happening, given that B has occurred. Here, B is the evidence and A is the hypothesis. The assumption made here is that the predictors/features are independent. That is, the presence of one particular feature does not affect the other. Hence it is called naive.

7.Error Analysis and Comparison:

7.1KNN

a) Bag of Words (BOW) transformation:



```
              precision    recall  f1-score   support

           0       0.65      0.72      0.68        39
           1       0.49      0.75      0.59        44
           2       0.94      0.80      0.87        41
           3       0.73      0.69      0.71        35
           4       1.00      0.51      0.68        41

    accuracy                           0.69       200
   macro avg       0.76      0.69      0.70       200
weighted avg       0.76      0.69      0.70       200
```

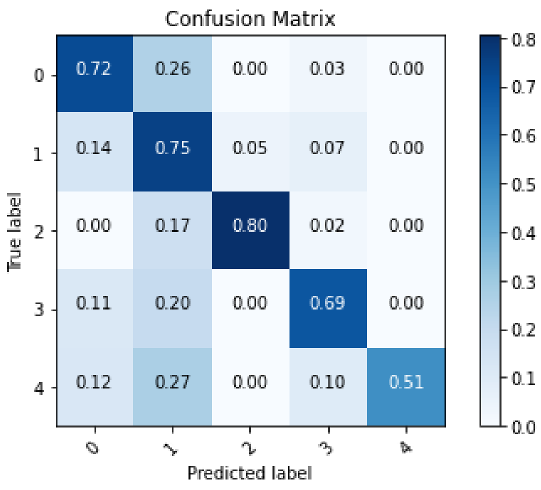


Figure 6 a) Error Analysis & Confusion Matrix of KNN algorithm with BOW transformation (K=3)

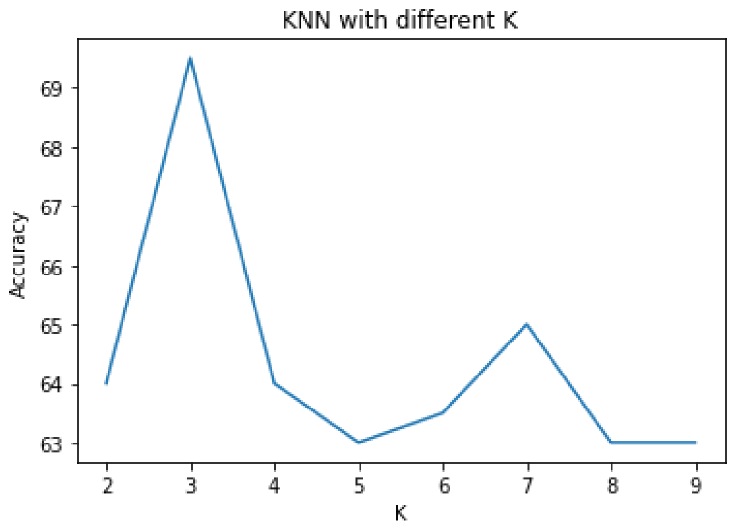
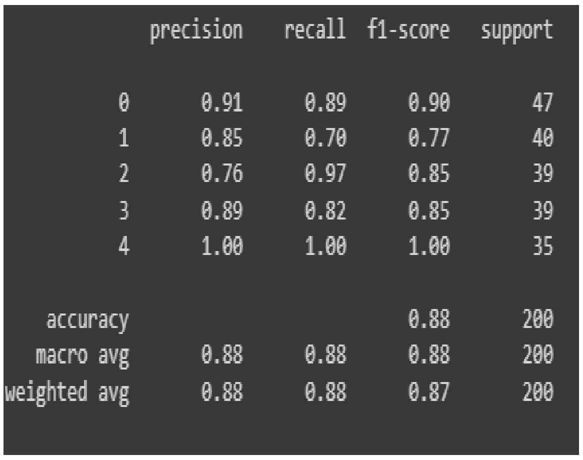


Figure 6 b) Accuracy scores of KNN algorithm with BOW transformation plotted against various values of K

b) TF-IDF transformation:



```
              precision    recall  f1-score   support

           0       0.91      0.89      0.90        47
           1       0.85      0.70      0.77        40
           2       0.76      0.97      0.85        39
           3       0.89      0.82      0.85        39
           4       1.00      1.00      1.00        35

    accuracy                           0.88       200
   macro avg       0.88      0.88      0.88       200
weighted avg       0.88      0.88      0.87       200
```

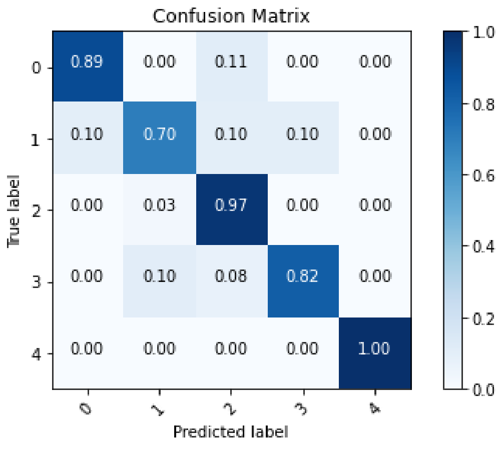


Figure 7 a) Error Analysis & Confusion Matrix of KNN algorithm with TF-IDF transformation (K=7)

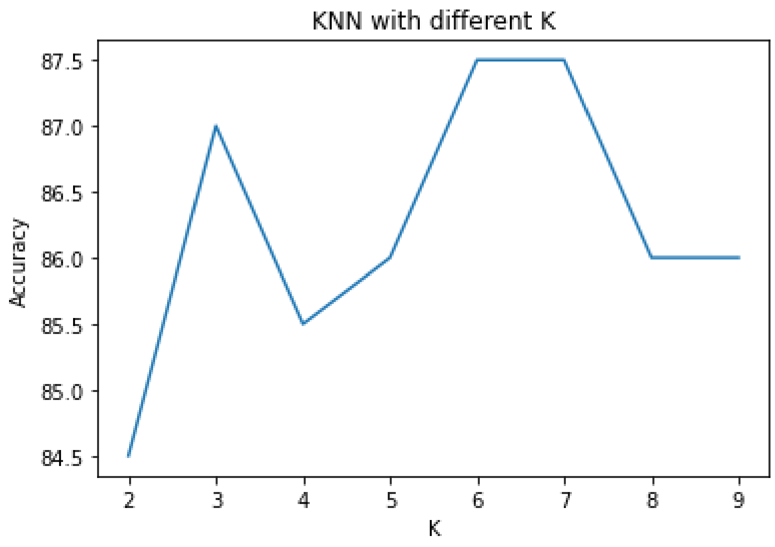
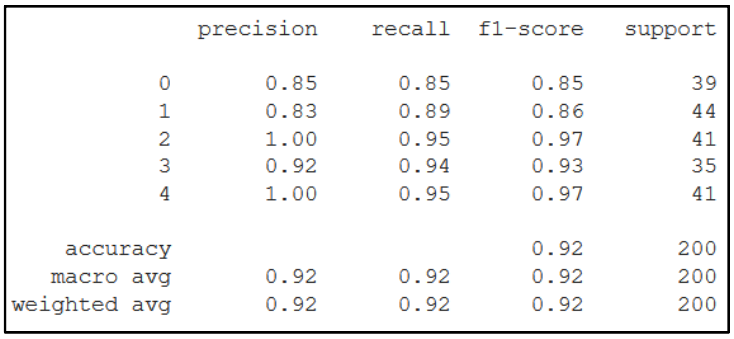


Figure 7 b) Accuracy scores of KNN algorithm with TF-IDF transformation plotted against various values of K

7.2 SVM

a) BOW Transformation



```
              precision    recall  f1-score   support

           0       0.85      0.85      0.85        39
           1       0.83      0.89      0.86        44
           2       1.00      0.95      0.97        41
           3       0.92      0.94      0.93        35
           4       1.00      0.95      0.97        41

    accuracy                           0.92       200
   macro avg       0.92      0.92      0.92       200
weighted avg       0.92      0.92      0.92       200
```

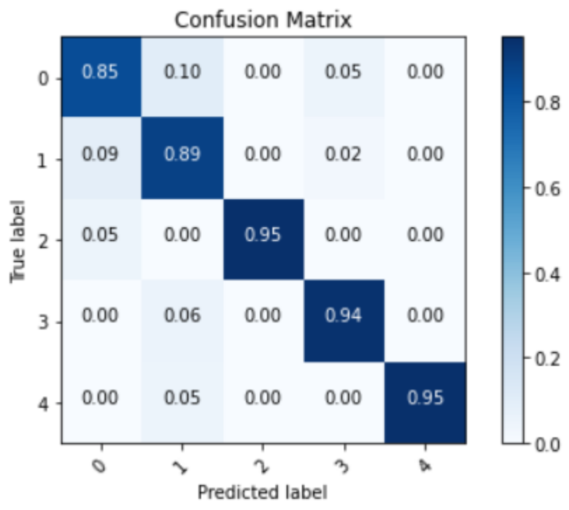
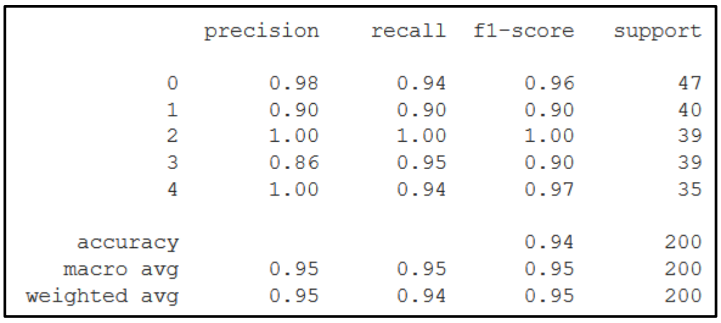


Figure 8 a) Error Analysis & Confusion Matrix of SVM using BOW C=1.0, kernel=`linear`

b) TF-IDF Transformation



```
              precision    recall  f1-score   support

           0       0.98      0.94      0.96        47
           1       0.90      0.90      0.90        40
           2       1.00      1.00      1.00        39
           3       0.86      0.95      0.90        39
           4       1.00      0.94      0.97        35

    accuracy                           0.94       200
   macro avg       0.95      0.95      0.95       200
weighted avg       0.95      0.94      0.95       200
```

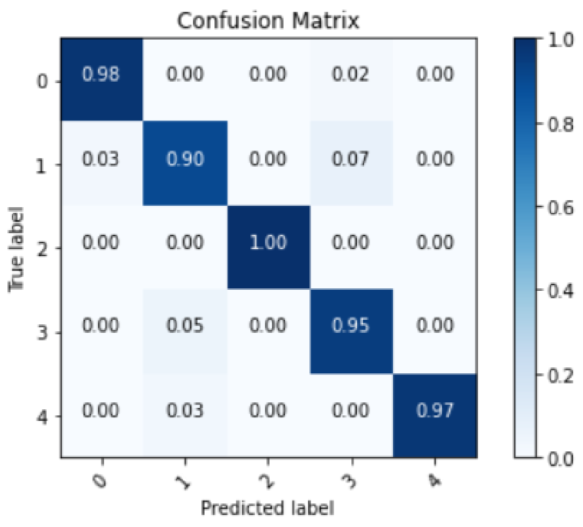
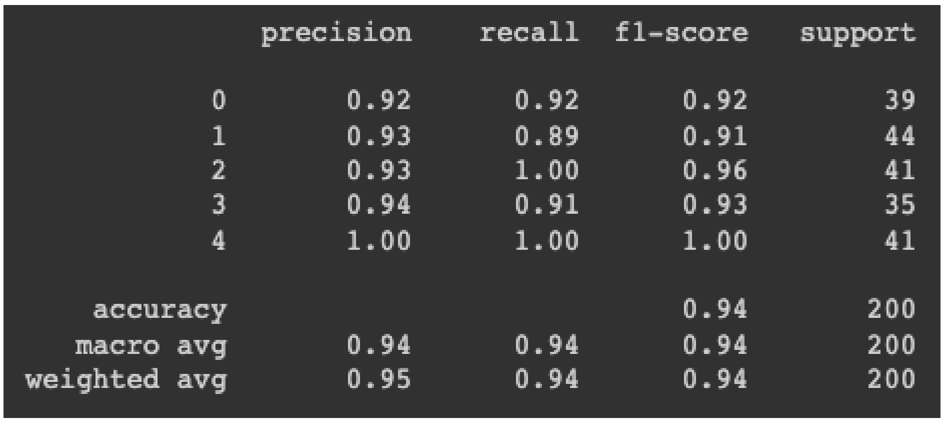


Figure 8 b) Error Analysis & Confusion Matrix of SVM using TF-IDF C=0.6, kernel=`linear`

7.3 Naive Bayes:



```
              precision    recall  f1-score   support

           0       0.92      0.92      0.92        39
           1       0.93      0.89      0.91        44
           2       0.93      1.00      0.96        41
           3       0.94      0.91      0.93        35
           4       1.00      1.00      1.00        41

    accuracy                           0.94       200
   macro avg       0.94      0.94      0.94       200
weighted avg       0.95      0.94      0.94       200
```

Figure 9 a) Error Analysis using NaiveBayes with TF-IDF

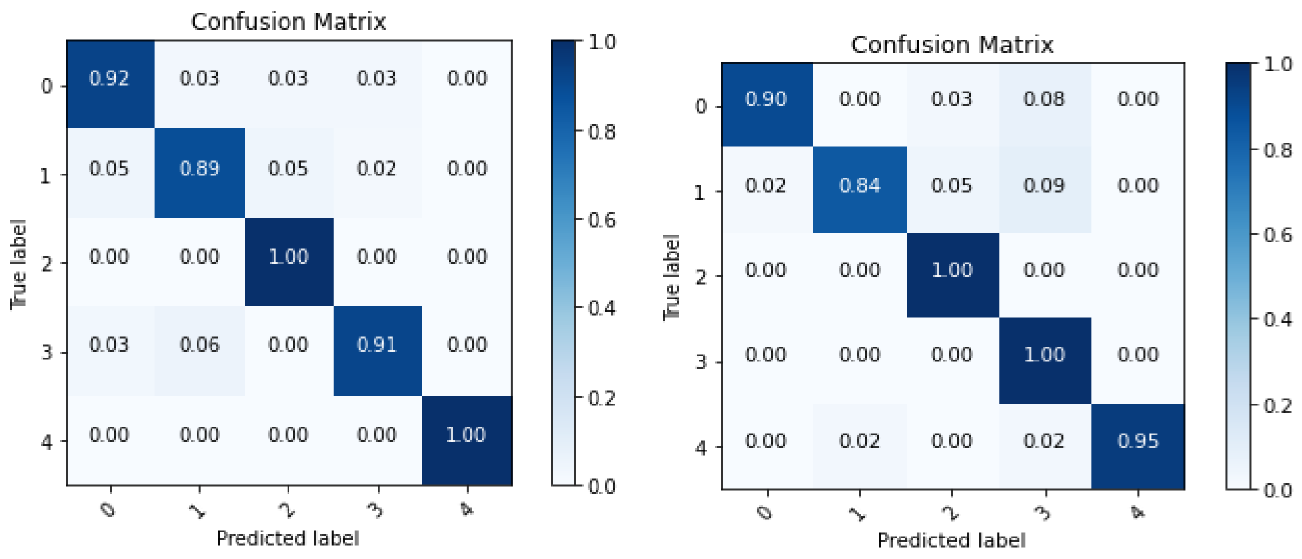


Figure 9 b) Confusion Matrix of NaiveBayes with BOW(left) and TF-IDF(right)

8.Conclusion

The following points were concluded for this project to classify the books.

- SVM among the three approaches gave the most accurate results and thus is chosen as the champion model.
- Accuracy drops after decreasing the words per partition.Thus, for better accuracy
- The preprocessing of words i.e removing numbers varies the accuracy.
- Changing bias and variability affects accuracy.
- TF-IDF gave better accuracy than Bag of Words.

- Books 0,1 and 3 are harder to classify.

References:

[1]https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c

[2]https://www.analyticsvidhya.com/blog/2020/02/quick-introduction-bag-of-words-bow-tf-idf/

[3]https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

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