Documentation

Data:

Our business case is to predict most valuable travel book based on user preference

We choose 5 books from Gutenberg library and they are:

1 – “A London Life” which was written by the author “Henry James”, we label it in our dataframe as “c”, you can find the book in the following link:

https://www.gutenberg.org/files/25500/25500-8.txt

2 – “Malay Magic” which was written by the author “Walter William Skeat”, we label it in our dataframe as “d”, you can find the book in the following link:

https://www.gutenberg.org/files/47873/47873-8.txt

3 – “Chicago by day and night” which was written by anonymous author, we label it in our dataframe as “b”, you can find the book in the following link:

https://www.gutenberg.org/files/70675/70675-0.txt

4 – “The Private Life of the Romans” which was written by the author “Harold Whetstone Johnston”, we label it in our dataframe as “e”, you can find the book in the following link:

https://www.gutenberg.org/files/40549/40549-0.txt

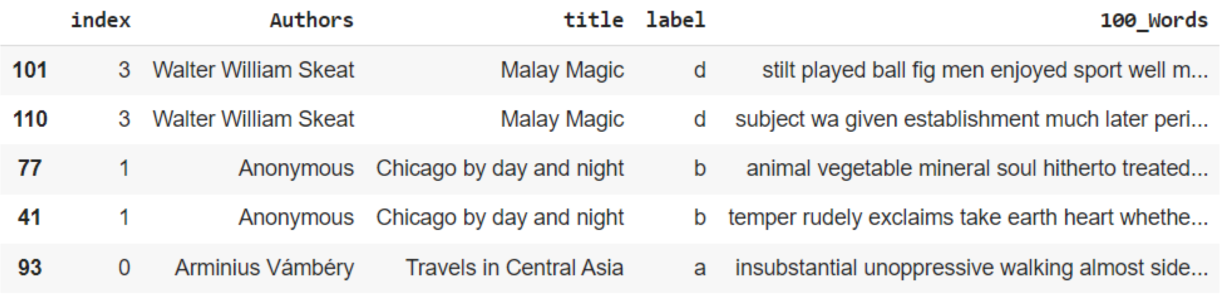
5 – “Travels in Central Asia” which was written by the author “Arminius Vámbéry”, we label it in our dataframe as “a”, you can find the book in the following link:

https://www.gutenberg.org/files/41751/41751-8.txt

In data preprocessing we remove stop words that don’t really signify any importance and don’t distinguish the books, then we did lemmatization to convert the words to its meaningful base form, we also did lower casing to convert the word to its lower case for simplicity.

After that we create samples of 200 documents (200 partition), each partition has 100 words

The output of the preprocessing process is:

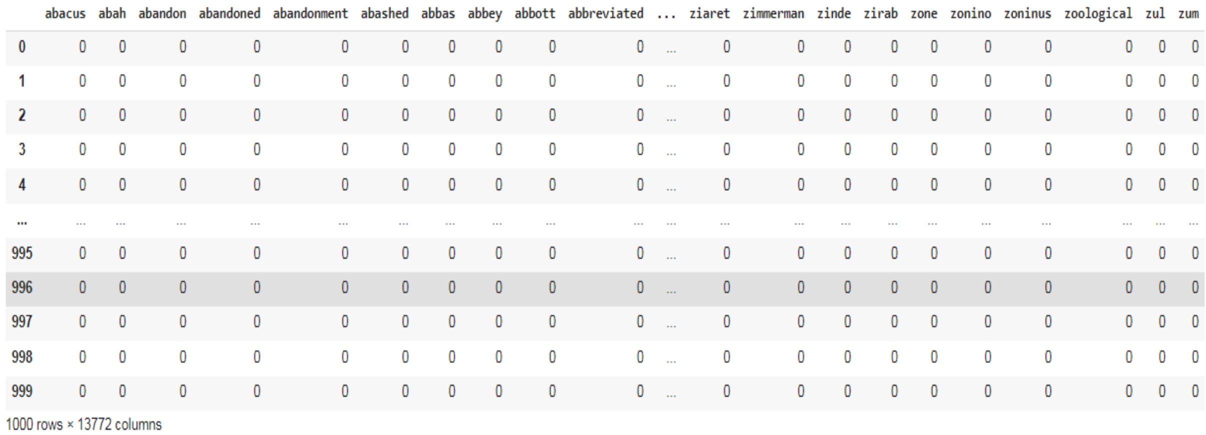


	index	Authors	title	label	100_Words
101	3	Walter William Skeat	Malay Magic	d	stilt played ball fig men enjoyed sport well m...
110	3	Walter William Skeat	Malay Magic	d	subject wa given establishment much later peri...
77	1	Anonymous	Chicago by day and night	b	animal vegetable mineral soul hitherto treated...
41	1	Anonymous	Chicago by day and night	b	temper rudely exclaims take earth heart whethe...
93	0	Arminius Vámbéry	Travels in Central Asia	a	insubstantial unoppressive walking almost side...

Feature Engineering:

Feature engineering is the process of transforming raw data into features that can be used to train machine learning models. One common technique for feature engineering is to use bag-of-words (BOW) and term frequency-inverse document frequency (TF-IDF) representations we use in our model:

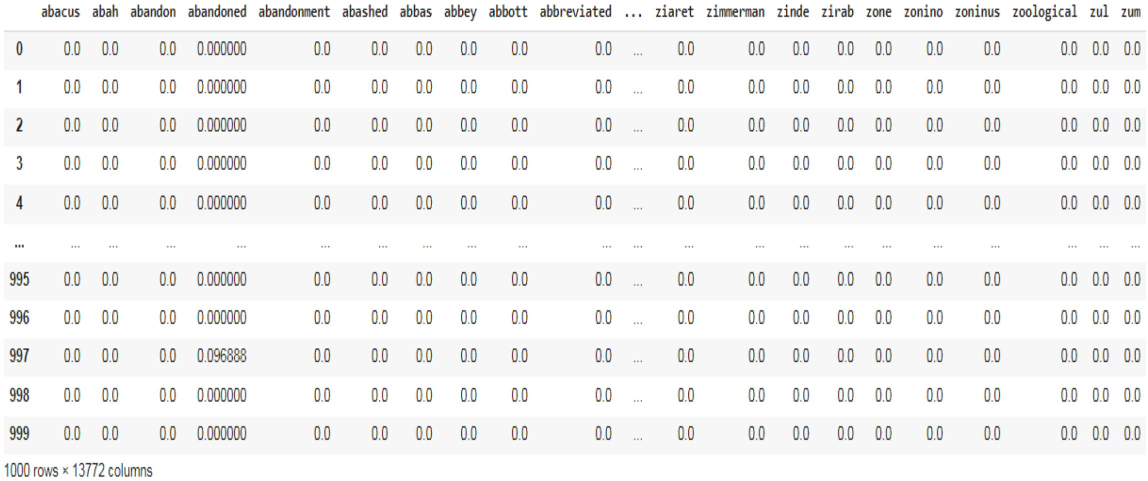
1- BOW: creates a matrix of word counts for each document in a corpus. The CountVectorizer() function from the Scikit-learn library can be used to perform this transformation. The output of the code:



	abacus	abah	abandon	abandoned	abandonment	abashed	abbas	abbey	abbott	abbreviated	...	ziaret	zimmerman	zinde	zirab	zone	zonino	zoninus	zoological	zul	zum
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
995	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
996	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
997	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
998	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
999	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

1000 rows × 13772 columns

2-TF-IDF: is a technique that weights the importance of each word in a document based on its frequency in the document and its frequency in the entire corpus. We used TF-IDF after BOW for feature engineering because it takes into account the importance of each word in the document and in the corpus, which can lead to better performance in machine learning models. The output of the code:

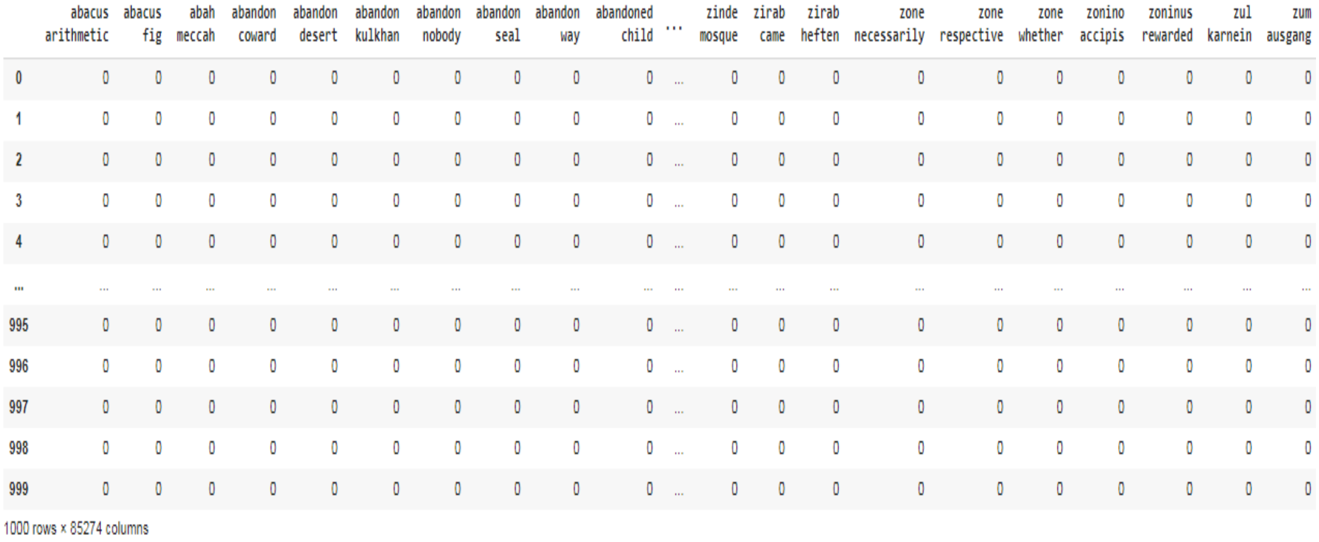


	abacus	abah	abandon	abandoned	abandonment	abashed	abbas	abbey	abbott	abbreviated	...	ziaret	zimmerman	zinde	zirab	zone	zonino	zoninus	zoological	zul	zum
0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
995	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
996	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
997	0.0	0.0	0.0	0.096888	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
998	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
999	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

1000 rows × 13772 columns

3- N-GRAM:

We also used n-grams to capture the context of the words. N-grams are contiguous sequences of n items from a given sample of text. For example, using bigrams (n=2) would result in a representation that captures pairs of adjacent words. Using n-grams can improve the performance of text-based models by capturing some of the local context around each word.



	abacus arithmetic	abacus fig	abah meccah	abandon coward	abandon desert	abandon kulkhan	abandon nobody	abandon seal	abandon way	abandoned child	...	zinde mosque	zirab came	zirab heften	zone necessarily	zone respective	zone whether	zonino accipis	zoninus rewarded	zul karnein	zum ausgang
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
995	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
996	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
997	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
998	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
999	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

1000 rows × 85274 columns

Modeling:

1- Benefits of Support Vector Machine: SVM performs reasonably well when there is a large degree of class separation.

In large dimensional spaces, SVM performs better.

If there are more dimensions than samples, SVM works well in certain situations. SVM uses relatively little memory.

2- The data itself serves as a model that will serve as the basis for future predictions, therefore KNN modeling does not include a training phase. As a result, it is very time-efficient when improvising for random modeling on the available data.

KNN is fairly simple to build because all that needs to be calculated is the distance between various points using information about various attributes;

this distance may be estimated using distance formulas like Euclidian or Manhattan distance.

New data can be uploaded at any time because the model won't be affected since there is no training period.

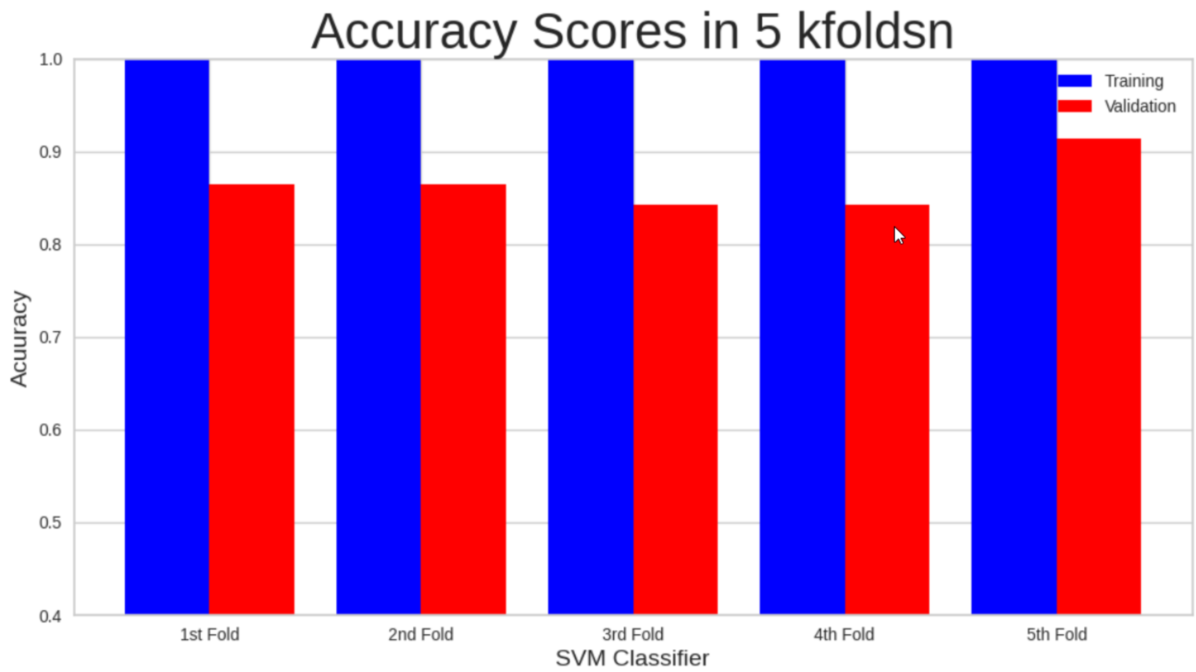
3- When training a machine learning model, the optimization approach gradient descent is applied. It uses a convex function as its foundation and iteratively adjusts its parameters to minimize a given function to its local minimum.

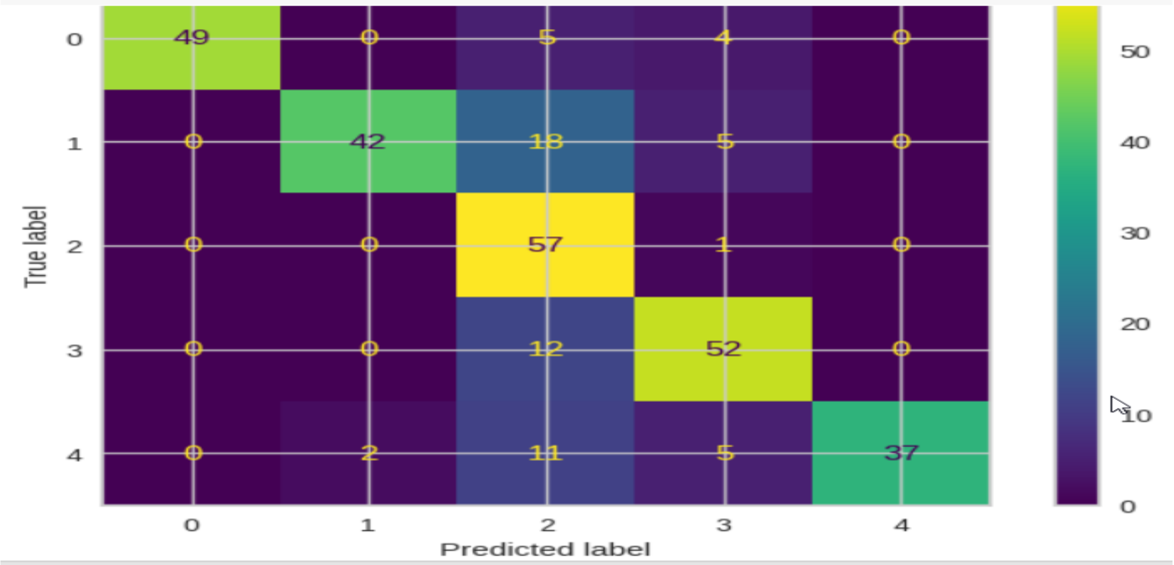
4- An ensemble learning technique called random forests or random choice forests works by building a large number of decision trees during the training phase. The class that the majority of the trees chose is the output of the random forest for classification problems.

5- The Bayes' Theorem is used by naive Bayes, which also presupposes that each predictor is independent. To put it another way, this classifier makes the assumption that the presence of one specific feature in a class has no bearing on the presence of another.

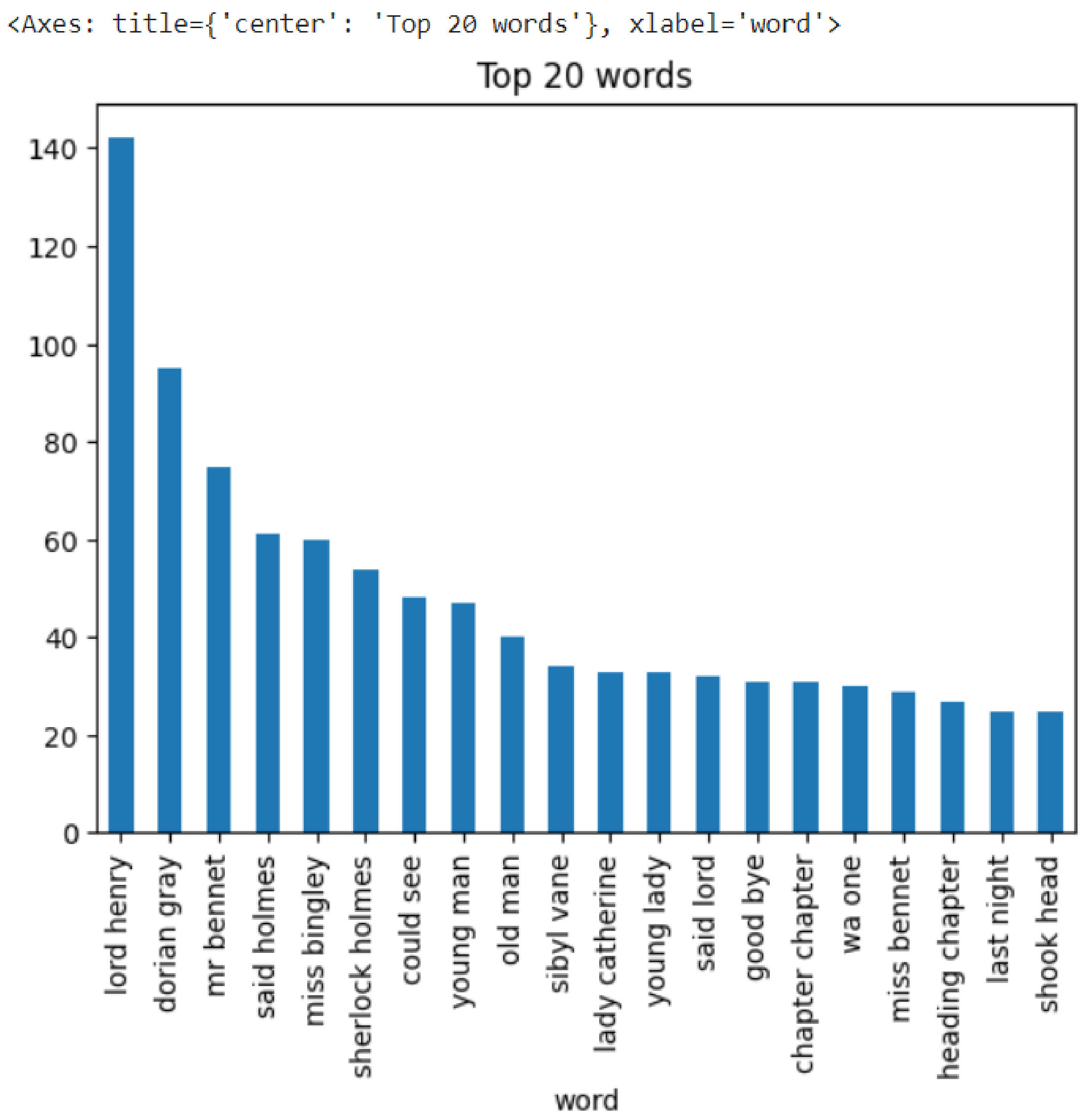
Error analysis:

- When using n-gram we notice that there is overfitting (Train accuracy was too high and validation accuracy was low) and we infer this to Insufficient Training Data: When the training data is too small, the model may memorize the training data instead of learning general patterns. So this can lead to overfitting. example for this is when using SVM Classifier Based on N-Gram like in the figure below:


Accuracy Scores in 5 kfoldsn
Training
Validation
Acuuracy
1.0
0.9
0.8
0.7
0.6
0.5
0.4
1st Fold
2nd Fold
3rd Fold
4th Fold
5th Fold
SVM Classifier


True label
0
1
2
3
4
49 0 5 4 0
0 42 18 5 0
0 0 57 1 0
0 0 12 52 0
0 2 11 5 37
0
1
2
3
4
Predicted label
50
40
30
20
10
0

- The figure below show the top 20 most frequent bigrams in the corpus and that's why the model couldn't fit well:



```
<Axes: title={'center': 'Top 20 words'}, xlabel='word'>
```

Analysis of Bias and Variability:

- After we found out that there are many common words between "A London life" and "The Private Life of the Romans" we explained this as the main reason behind this that the two authors were Americans so that's why they used many common words

The model's threshold:

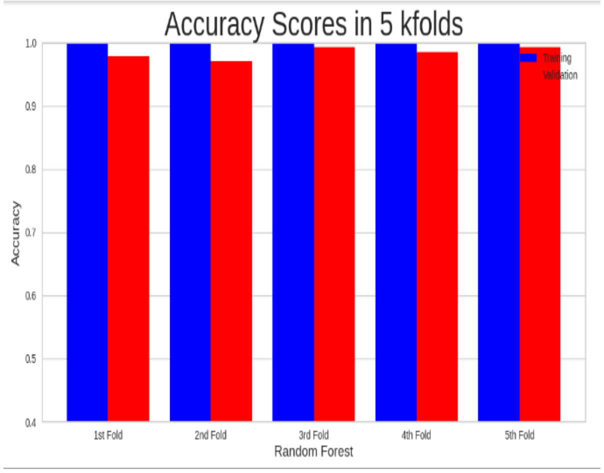
- After changing the SVM kernel parameter from "linear" to "poly" we found out that the accuracy changed from 0.993 to 0.5766 which is too low and this clarify that using a linear kernel is mandatory.

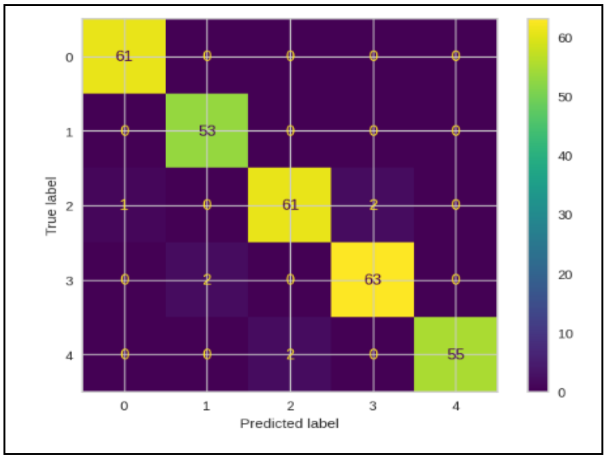
Testing Results:

1- Random Forest:

- Based on Bow:

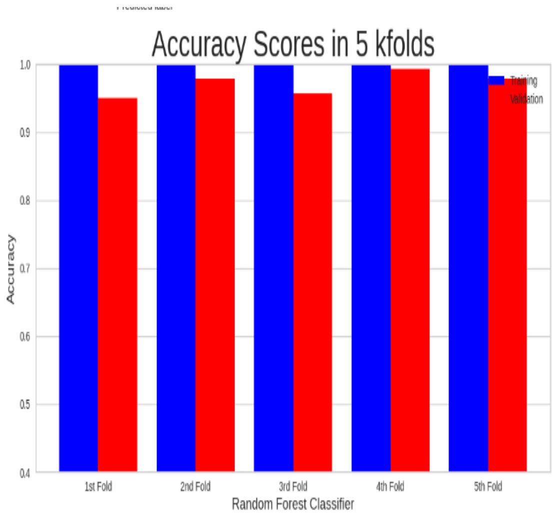
Accuracy: 0.9766666666666667

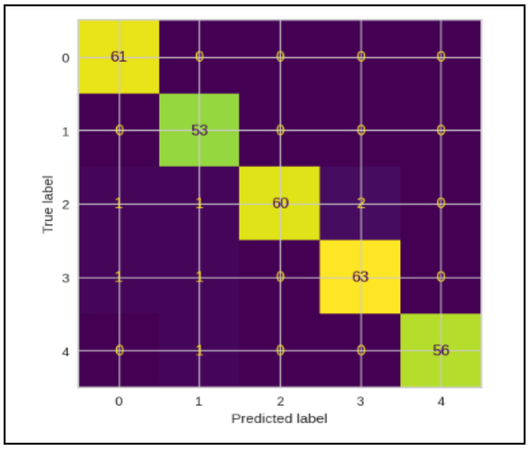




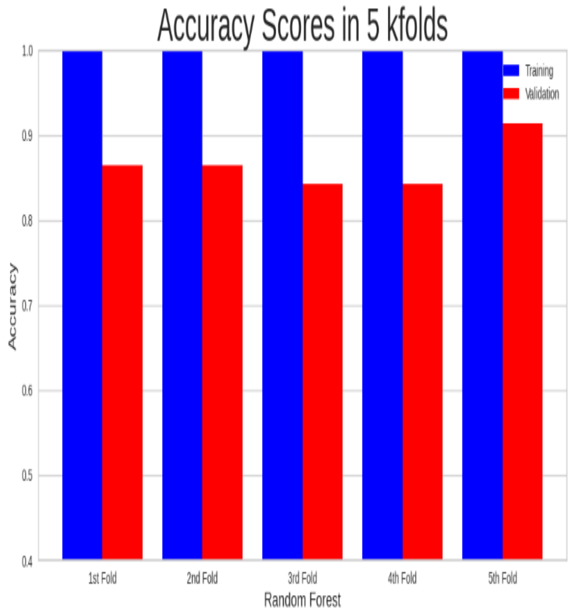
- Based on TF-IDF:

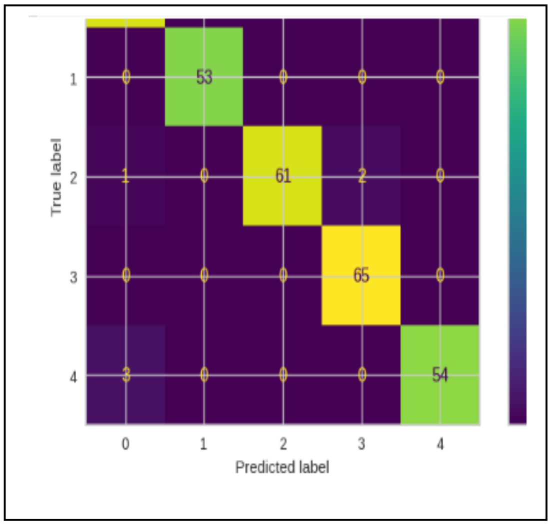
Accuracy: 0.9766666666666667





Based on n-gram:

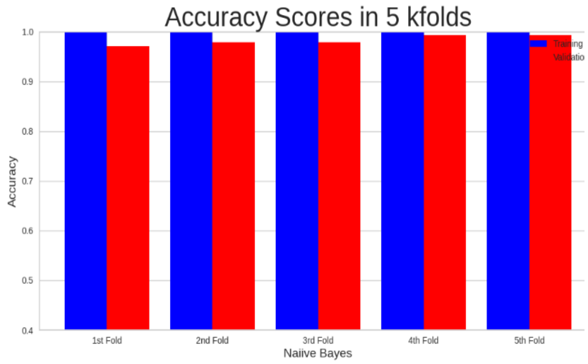


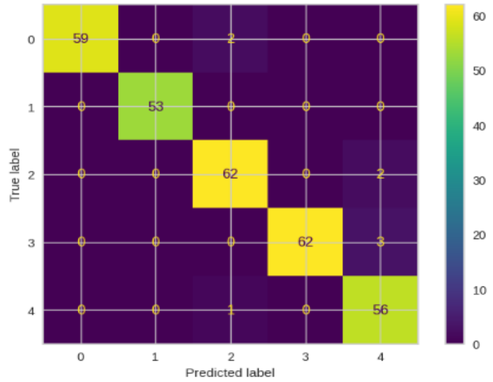


2- Naive Bayes:

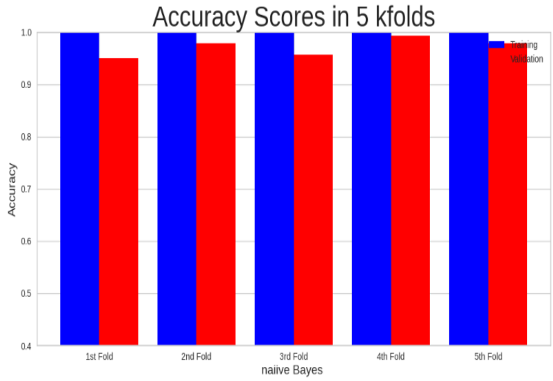
Based on Bow:

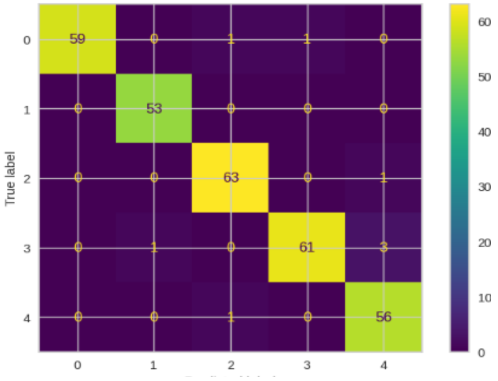
Accuracy: 0.9733333333333334



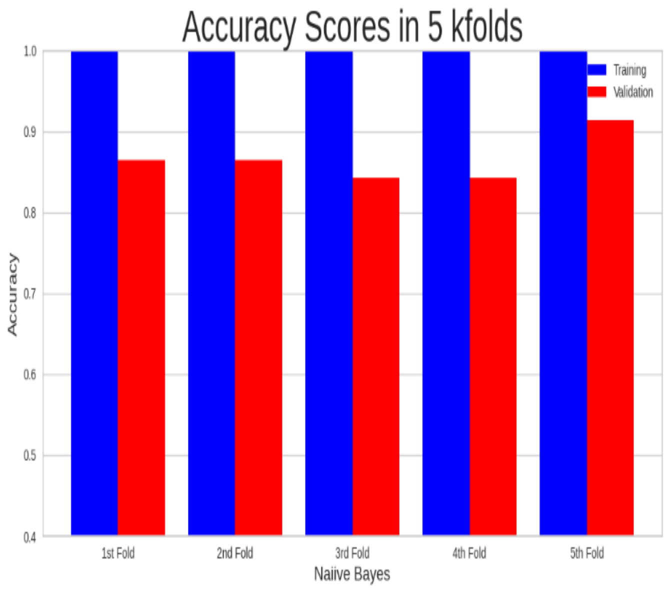


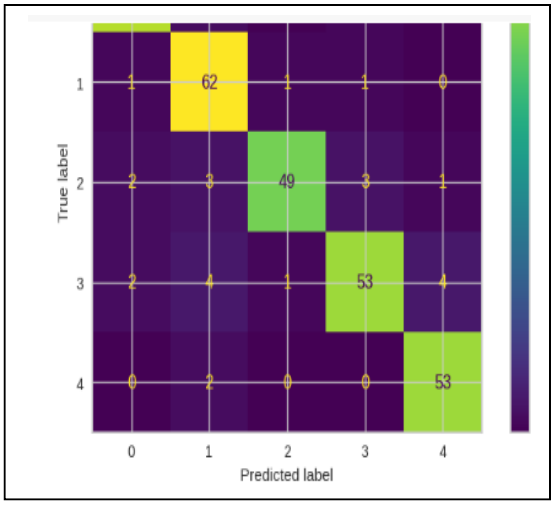
Based on TF-IDF:





- Based on n-gram:

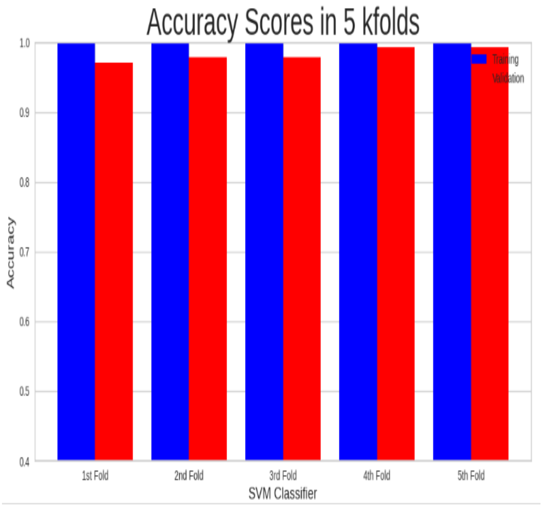


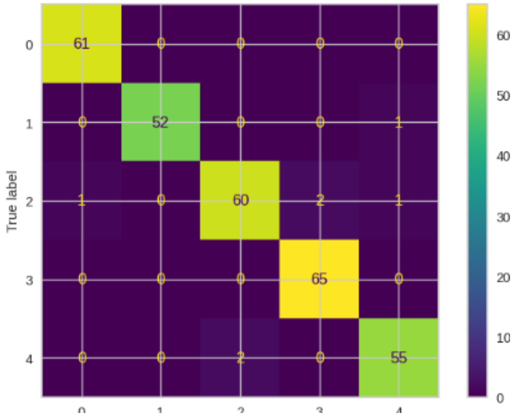


3- SVM Classifier:

Based on BOW:

Accuracy: 0.9766666666666667

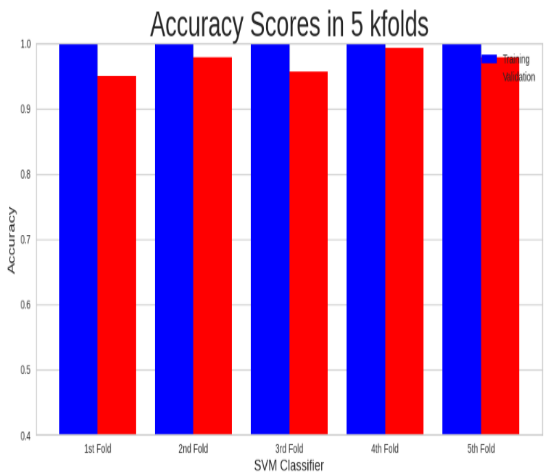


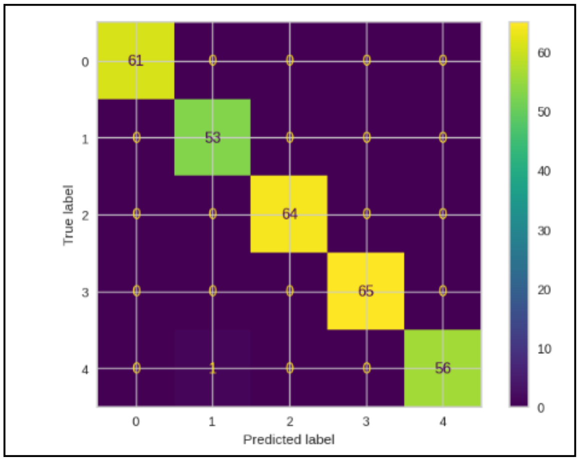


Based on TF-IDF:

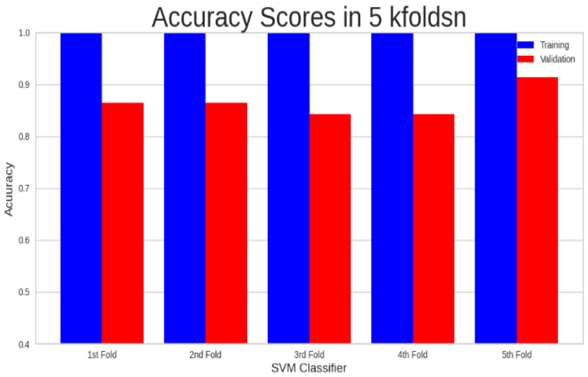
Accuracy: 0.9966666666666667

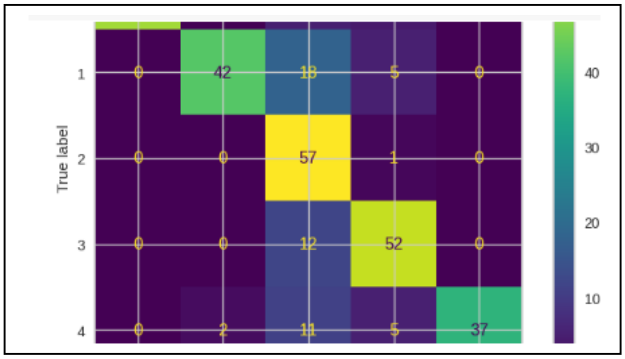
Precision Score: 0.9966666666666667





- Based on n-gram:

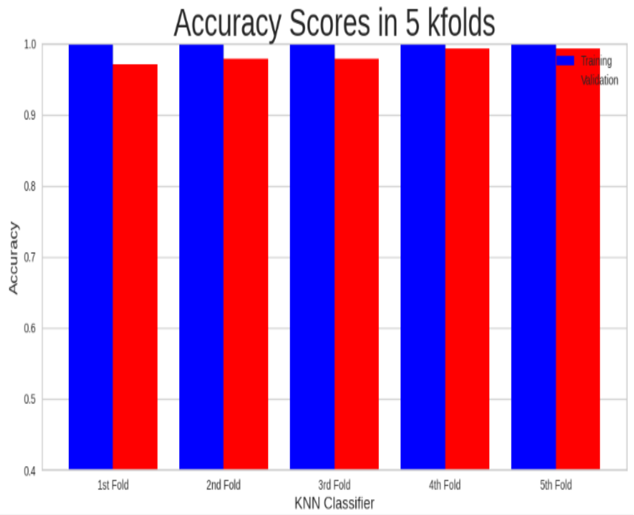


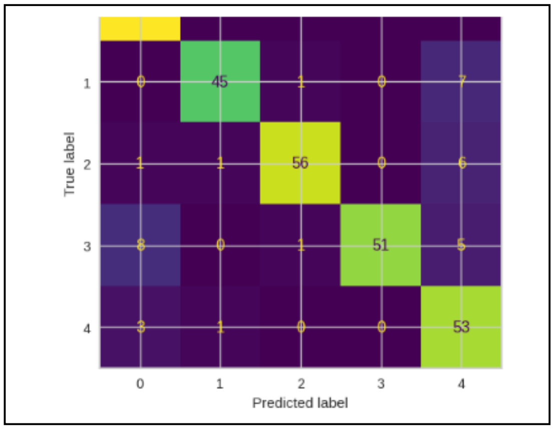


4- KNN Classifier:

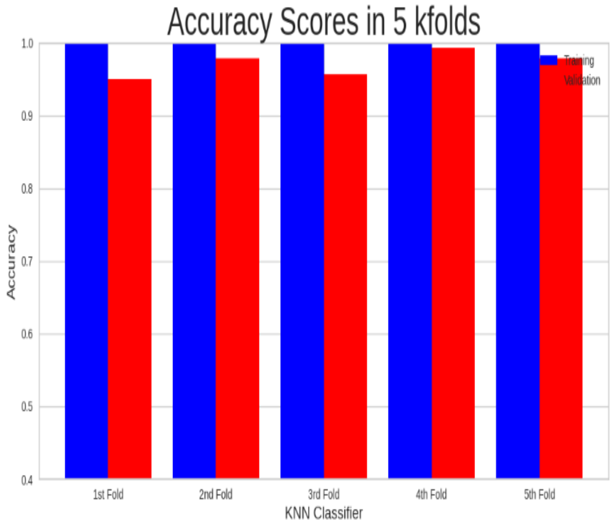
Based on Bow:

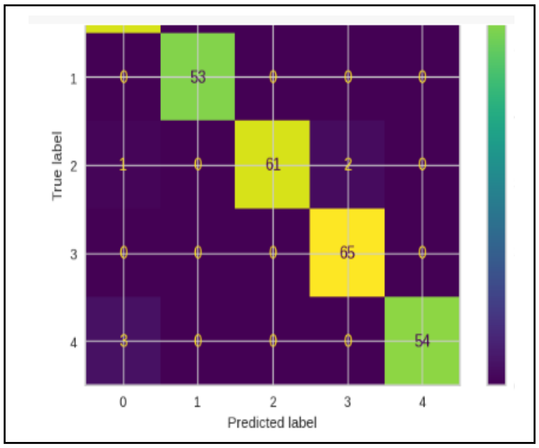
Accuracy: 0.8866666666666667



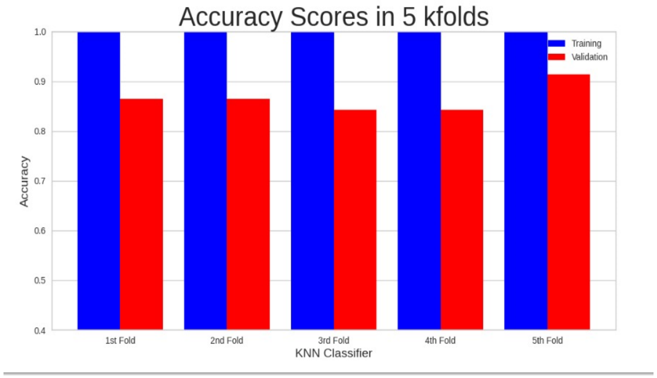


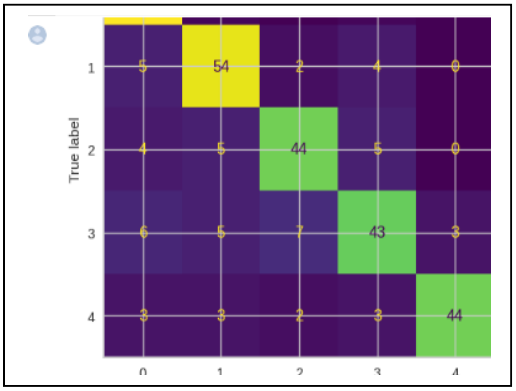
Based on TF-IDF:





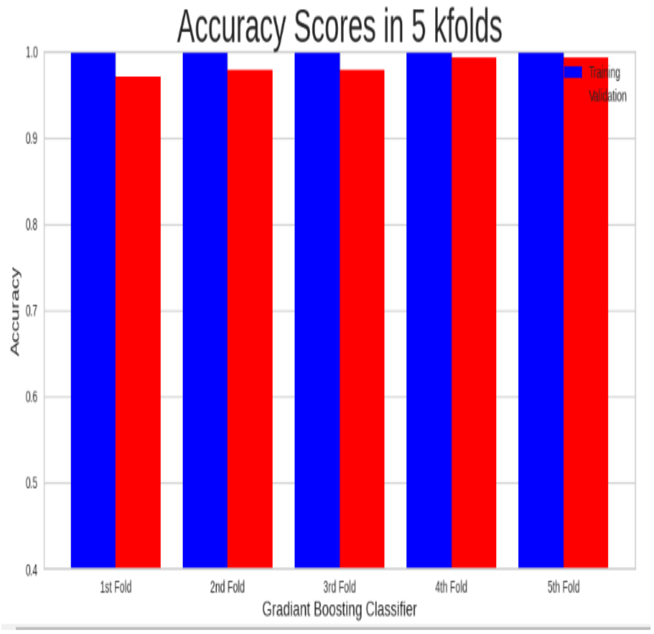
- Based on n-gram:

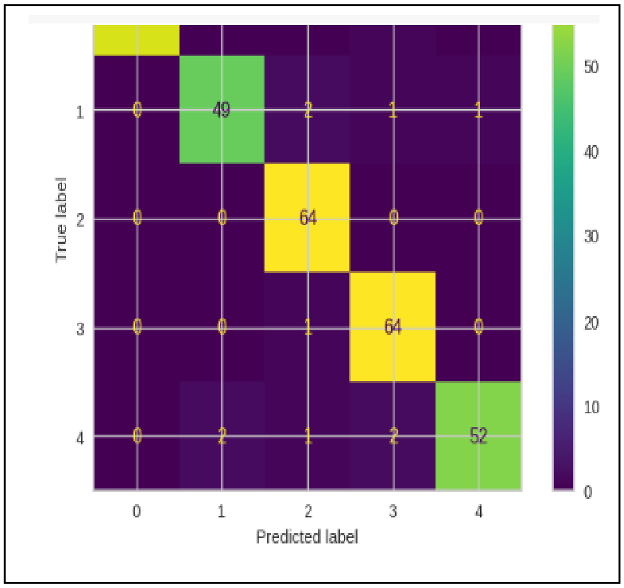




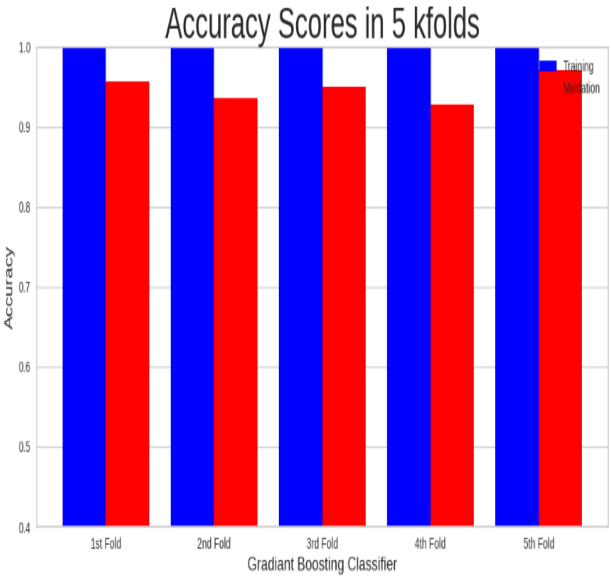
5-Gradiant Boosting Classifier:

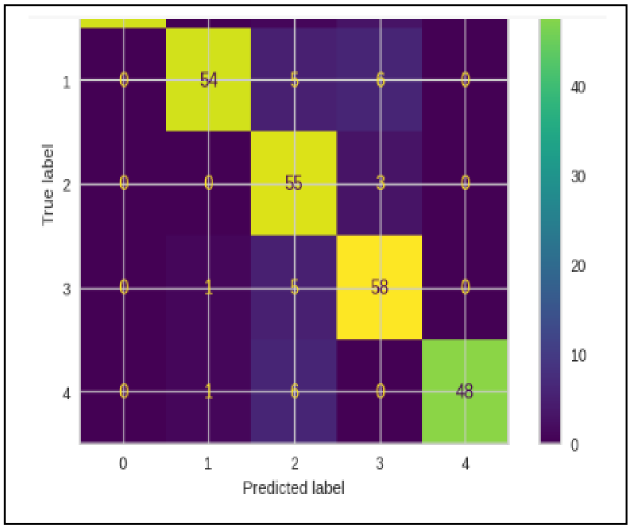
Based on BOW:



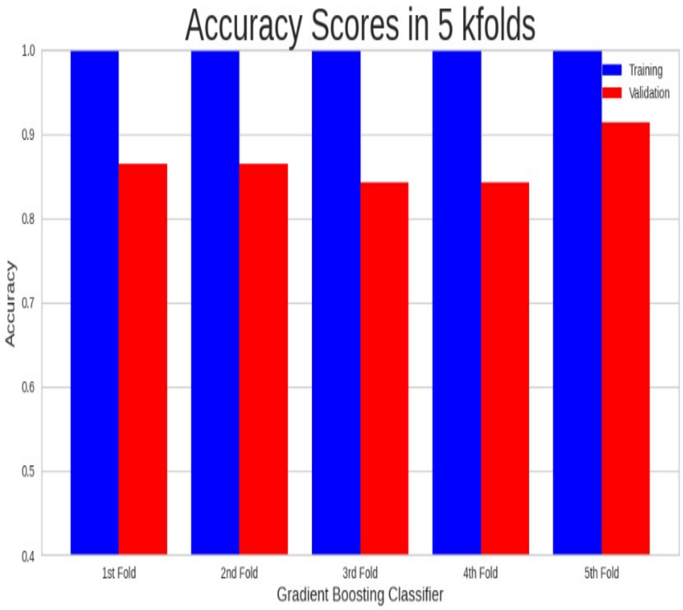


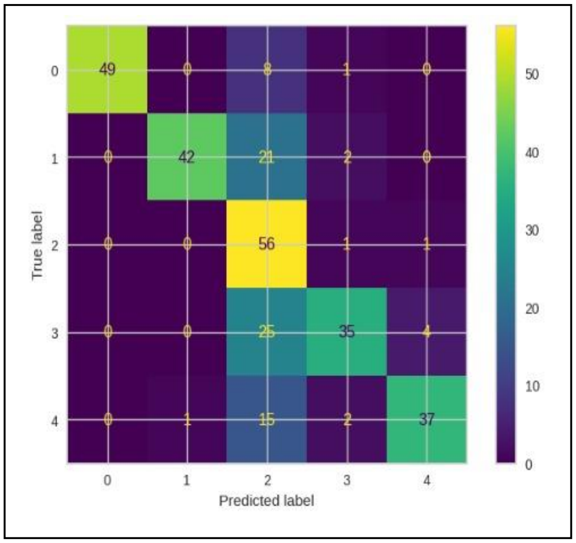
Based on TF-IDF:





- Based on n-gram:





Programs README:

- Installation: To use this program, you will need to install the required dependencies and libraries which had been installed in the first cell.
- Customization: Some libraries might not exist in your environment like yellowbrick for example so you need to install them through command-line to download the configuration files.
- Usage: Once installed, you can run the program and specify the classification task you would like to perform.