

**MACHINE LEARNING**

**Assessment Report**

**MOD006562**

**SID – 2118074**

**BSc Artificial Intelligence**

**Anglia Ruskin University**

**Computing & Technology Department**

**Date – 18/04/2023**

Contents

[What is Machine Learning? 4](#_Toc163812031)

[How does Machine Learning work? 4](#_Toc163812032)

[Introduction 5](#_Toc163812033)

[Objective & Scope 5](#_Toc163812034)

[Data Pre-processing, Preparation, & Analysis 5](#_Toc163812035)

[Vectorization of Data 26](#_Toc163812036)

[Preparing Data 28](#_Toc163812037)

[Train-Test Split 28](#_Toc163812038)

[Characteristics of Training set 28](#_Toc163812039)

[Characteristics of Testing set 28](#_Toc163812040)

[Cross Validation 28](#_Toc163812041)

[Training and Evaluating Models 29](#_Toc163812042)

[1. MODEL - Random Forest Classification 30](#_Toc163812043)

[1.1. Random Forest Classification - HASHING VECTORIZATION 30](#_Toc163812044)

[1.2. Random Forest Classification – TF-IDF 32](#_Toc163812045)

[1.3. Random Forest Classification – BAG OF WORDS (BOW) 34](#_Toc163812046)

[2. MODEL – Multinomial NB 36](#_Toc163812047)

[2.1. Multinomial NB - HASHING VECTORIZATION 36](#_Toc163812048)

[2.2. Multinomial NB - TF-IDF 38](#_Toc163812049)

[2.3. Multinomial NB - BAG OF WORDS (BOW) 40](#_Toc163812050)

[3. MODEL – Logistic Regression 42](#_Toc163812051)

[3.1. Logistic Regression - HASHING VECTORIZATION 42](#_Toc163812052)

[3.2. Logistic Regression - TF-IDF 44](#_Toc163812053)

[3.3. Logistic Regression - BAG OF WORDS 46](#_Toc163812054)

[4. MODEL – SVM 48](#_Toc163812055)

[4.1. SVM - HASHING VECTORIZATION 48](#_Toc163812056)

[4.2. SVM - TF-IDF 50](#_Toc163812057)

[4.3. SVM - BAG OF WORDS 52](#_Toc163812058)

[5. MODEL – KNN 54](#_Toc163812059)

[5.1. KNN – HASHING VECTORIZATION 54](#_Toc163812060)

[5.2. KNN – TF-IDF 56](#_Toc163812061)

[5.3. KNN – BAG OF WORDS 58](#_Toc163812062)

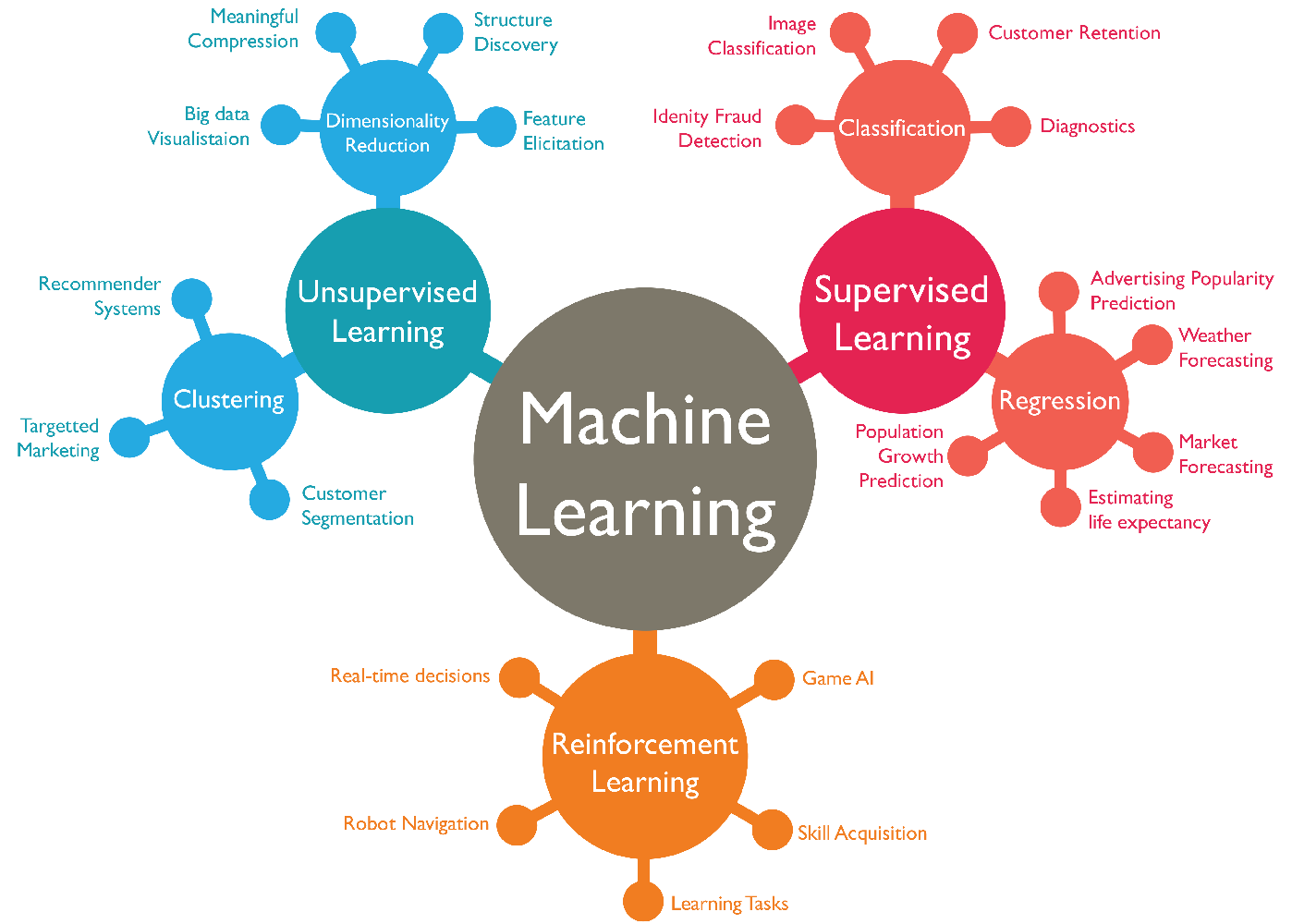
[Results/Evaluation Analysis 60](#_Toc163812063)

[Further Analysis and Methods 62](#_Toc163812064)

[Conclusion 65](#_Toc163812066)

[References 66](#_Toc163812067)

# What is Machine Learning?



Machine Learning (ML) is a branch of [Artificial Intelligence (AI)](https://www.ibm.com/topics/artificial-intelligence) and computer science that focuses on the using data and algorithms to enable AI to imitate the way that humans learn, gradually improving its accuracy.

# How does Machine Learning work?

A Decision Process: Machine learning algorithms are typically used to make predictions or classify data. Algorithm will generate an estimate about a pattern in some input data, which can be labelled or unlabelled. An error function is a function that evaluates the model's predictions. If there are known examples, an error function can compare them to determine the accuracy of the Model. A Model Optimisation Process: If the model can better fit the data points in the training set, weights are adjusted to reduce the difference between the known example and the model estimate. The algorithm will repeat the iterative "evaluate and optimise" process, updating weights autonomously until an accuracy threshold is met.

Machine learning models fall into three primary categories.

* Supervised Machine Learning
* Unsupervised Machine Learning
* Semi-Supervised Machine Learning
* Reinforcement Machine Learning

# Introduction

The aim of this report is to demonstrate the knowledge that the author have gained while working on the Machine Learning module project. This report logs the researches & experiments that the author have done on the topic Natural Language Processing (NLP), Classification & Vectorisation of the given Dataset. This process included different approaches in pre-processing data, training & testing data, classification of dataset, usage of different vectorization methods & Evaluation of the models to come up with a better accuracy percentage along with showcasing the methods and strategies that the author have learnt.

# Objective & Scope

The goal of the project is to implement and end to end machine learning pipeline based on the provided dataset of New Categories. This dataset consist of more than 200,000 new headlines that is categorised based on their topic into 41 different categories which have to analysed and use different techniques in machine learning to pre-process the data, conducting comprehensive NLP specific data preparations, training validation and testing of the data, statistical analysis of the dataset, usage of different vectorization techniques followed by evaluation of the results.

# Data Pre-processing, Preparation, & Analysis

The dataset provided for this assessment is highly imbalanced and intricate. The dataset have been explored and analysed thoroughly by the Author. The data have been pre-processed and prepared for classification and prediction. All the steps that were necessary to prepare the data have been demonstrated below. These steps includes –

* Loading the dataset
* Analysing the data
* Visualisation of the data
* Removing empty spaces
* Balancing the data
* Cleaning the data

**Data Exploration**

* **Loading the dataset**

Screenshot 2024-03-22 194555

Figure 1 Reading the dataset

The dataset was converted into a csv file before loading the dataset.

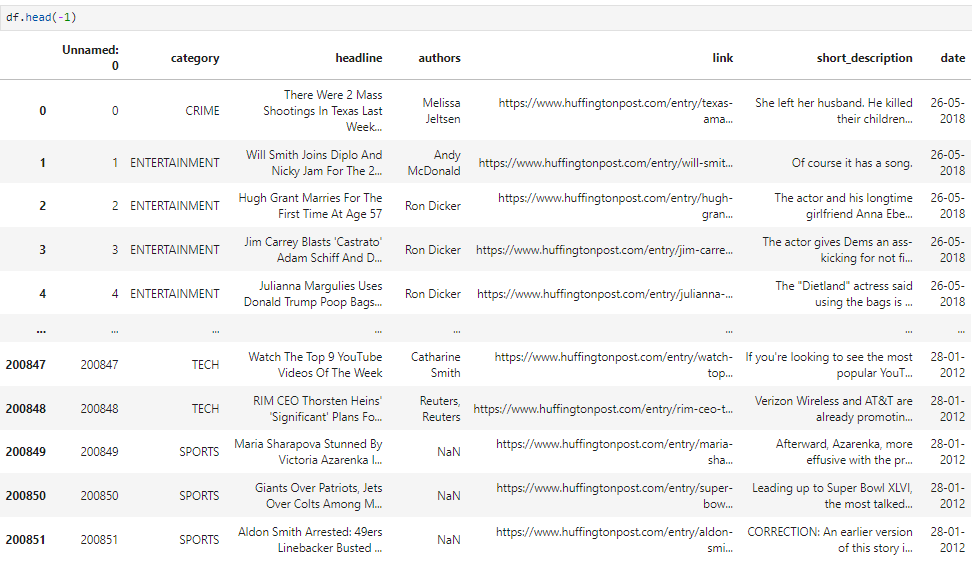


Figure 2 displaying the data

Displaying the dataset in-order to make sure that the data have been loaded successfully.

As the figure display ‘News\_Categories123.csv’ is a big dataset with almost 200,000 news articles from the year 2012 till 2018.

* **Analysing the data**





Figure 3 data info

Getting the information about the data.



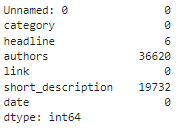


Figure 4 number of missing data

Checking if there is any null values in the dataset.



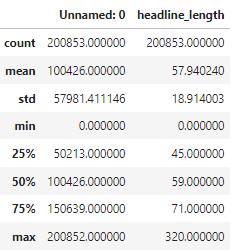


Figure 5 Description of the data

Summary statitics of the numeric data in the dataset.

(This is not relevent in this case)

C:\Users\acer\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Screenshot 2024-03-25 173504.png

Figure 6 Category count

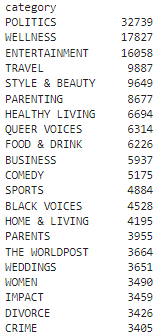
 

Figure 7 Number of data in each category

Checking the number of data in each category. As it is displayed below the dataset is highly imbalanced as the category ‘POLITICS’ has the most data which is 32,739 whereas the category ‘EDUCATION’ has the least amount of data 1,004. This clearly proves that the dataset is imbalanced. Visualisation of the amount of data for each category is presented below followed by the chart.

* **Visualisation of the data**



Figure 8 generating a bar plot

A well organised and easily understandable bar plot is generated to display the distribution of the data across categories.

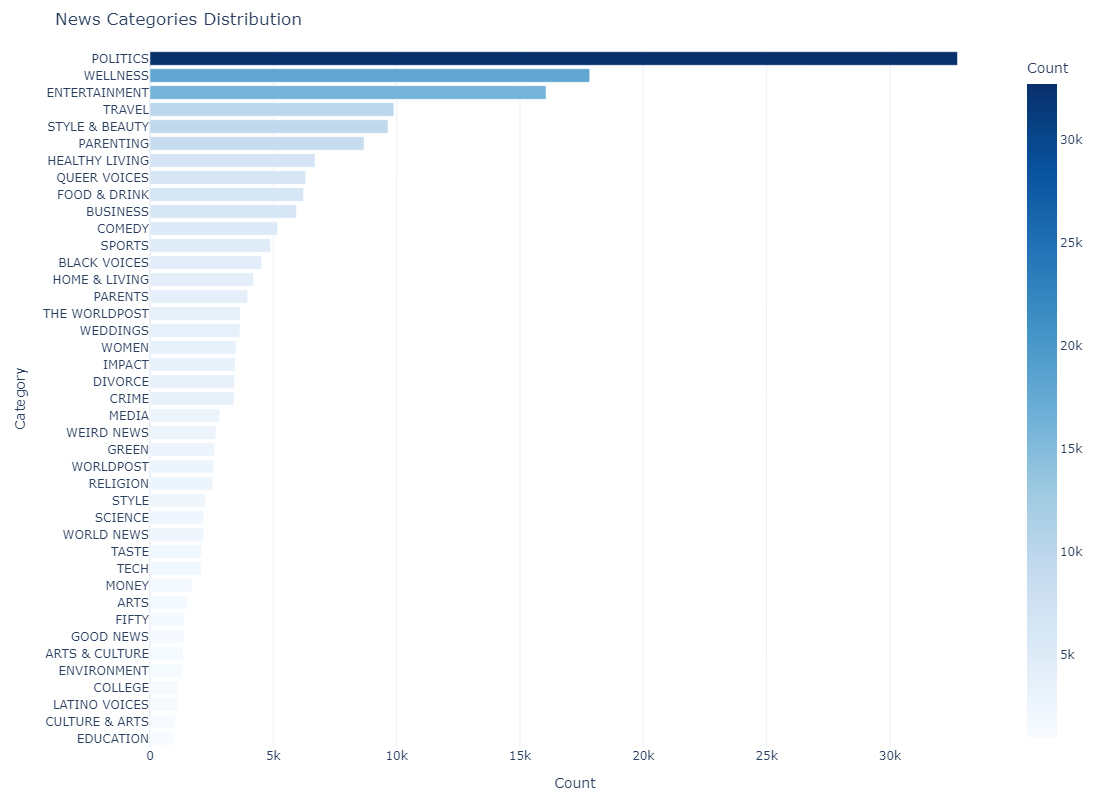


Figure 9 bar plot distribution of data across categories

It is clear that the data is imbalanced with Politics having the maximum number of articles followed by Wellness and Entertainment.

To carry on with the analysis, it is logical to concatenate 'headline' and 'short description' columns.

Generating a box plot to show the frequency of the headline string length.

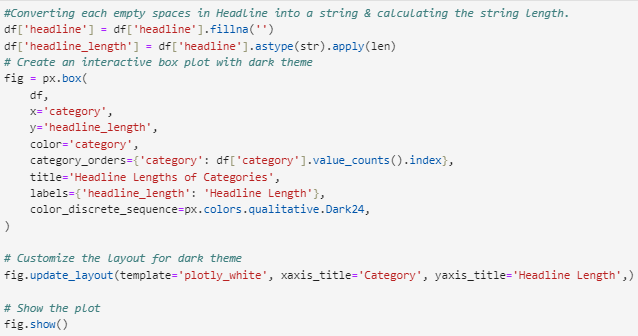


Figure 10 generating box plot

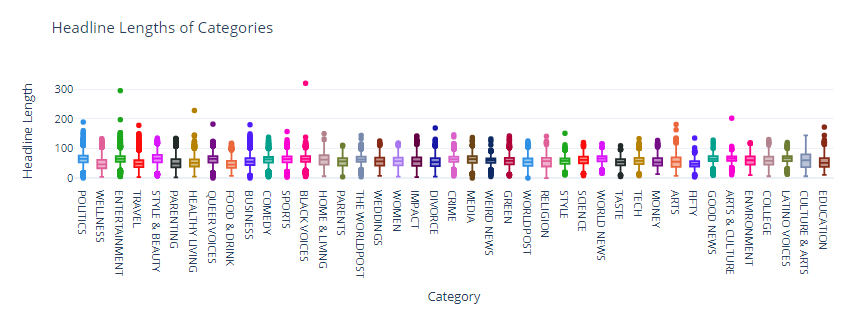


Figure 11 Box plot

This bar plot shows the string length of headline in each category. Most of the categories hold strings length which is around 150 – 200. Only a few of the categories have headlines length more than 200 characters.

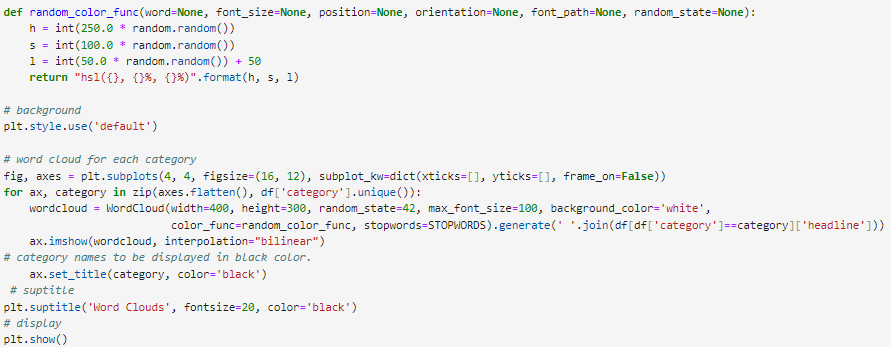


Figure 12 Generating word cloud



Figure 13 Word cloud

Word cloud representation of the most frequent words in the first 16 categories.

**What is Text pre-processing?**

Text pre-processing is a crucial step in Natural Language Processing (NLP). It involves cleaning and transforming raw text data into a format that is suitable for analysing and modelling.

* **Removing empty spaces**

Removing empty spaces is an important step while pre-processing. It reduces the noise in the text, radical memory efficiency improvement, makes the data more easily readable.



Figure 14 displays empty space



Figure 15 empty spaces in headline.



Figure 16 displays null value columns



Figure 17 short description with null value columns

The features ‘headline’, ‘short\_description’ & ‘authors’ have so many empty lines/ null values within the dataset as it is shown in here. All these empty spaces have to be deleted prior to the Natural Language Processing (NLP).

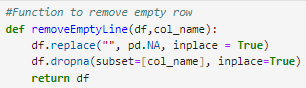


Figure 18 removing empty row



Figure 19 code to remove empty line

All the empty lines/ rows will be removes from this features.

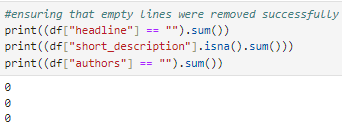


Figure 20 Ensuring if successful

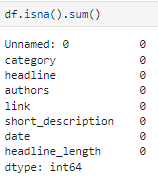


Figure 21 null value count

Ensuring there are no empty rows/ null values in the dataset before proceeding to the next stage.

* **Features –** ‘headline’ and ‘ short\_description’ contains all the relevant data for classification therefore proceeding with those two features
* **Deleting Unwanted Features -** Features such as ‘authors’, ‘links’, ‘date’ & ‘headline length’ doesn’t hold any useful data in order to perform text classification hence these feature have been dropped at this stage.

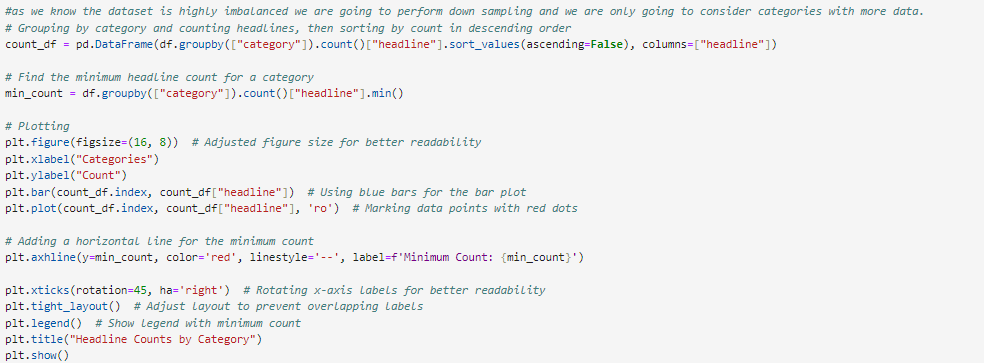


Figure 22 Visualization prior to balancing

This code generates a bar plot with the headline counts of each category just to demonstrate the imbalance once again with marking on the minimum count.

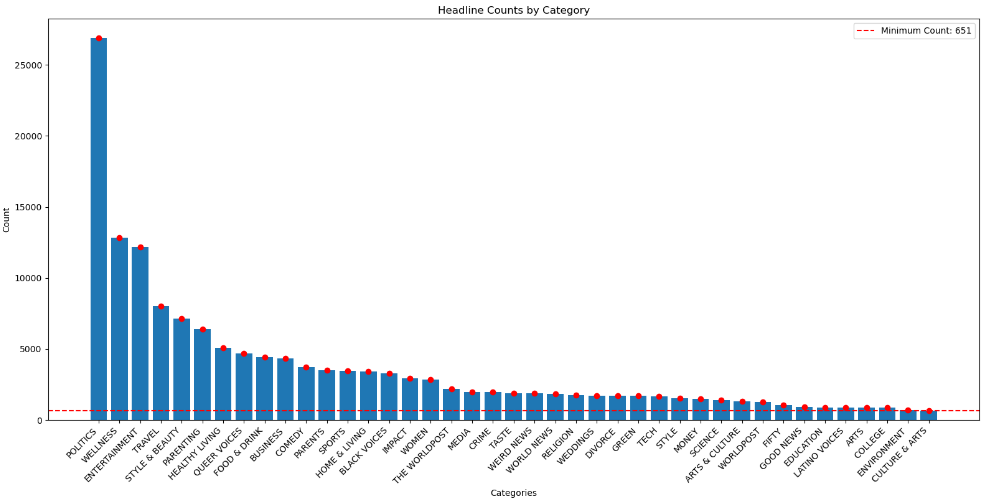


Figure 23 Headline count by category



Figure 24 category count

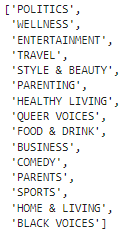


Figure 25 categories output

These are the categories with more than 3000 data.

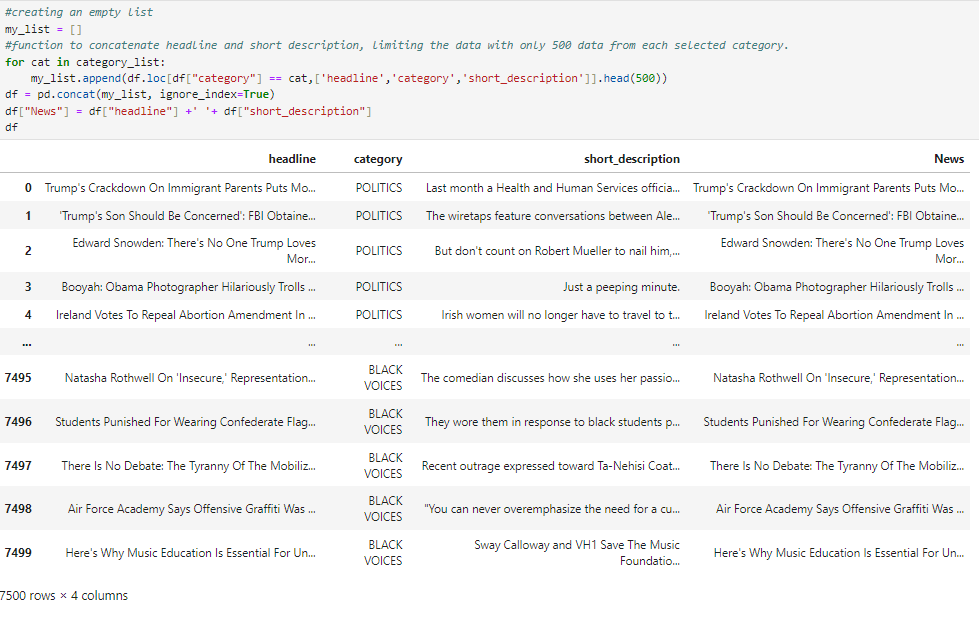


Figure 26 Merging relevant features

An empty list have been created followed by the concatenation of **‘headline’** and **‘short\_description’** creating a new feature ‘News’. The limitation of data (down sampling) have been done as processing large dataset demand high computational power which was a challenge for the author.

Since the data is extremely imbalanced, it is very difficult and tricky to implement simple classifiers that will give high accuracy. Thus for the purpose of this analysis, only categories with at least 3000 data and more will be considered. The data is also resampled to make sure all the categories as in equal proportion

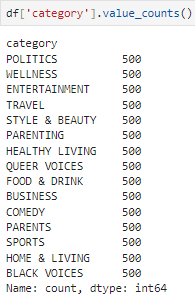


Figure 27 new count of data

These are the categories which had more than 3000 data and have been down sampled into 500 data.



Figure 28 dropping categories

These categories have been dropped as it possess similar features with other categories. Category ‘PARENTING’ is similar to ‘PARENTS’ & category ‘HEALTHY LIVING’ is similar to ‘HOME & LIVING’. These might confuse the algorithm hence being dropped from the dataset.

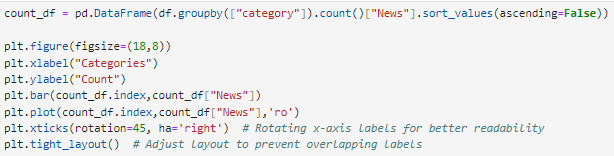


Figure 29 plotting all the categories

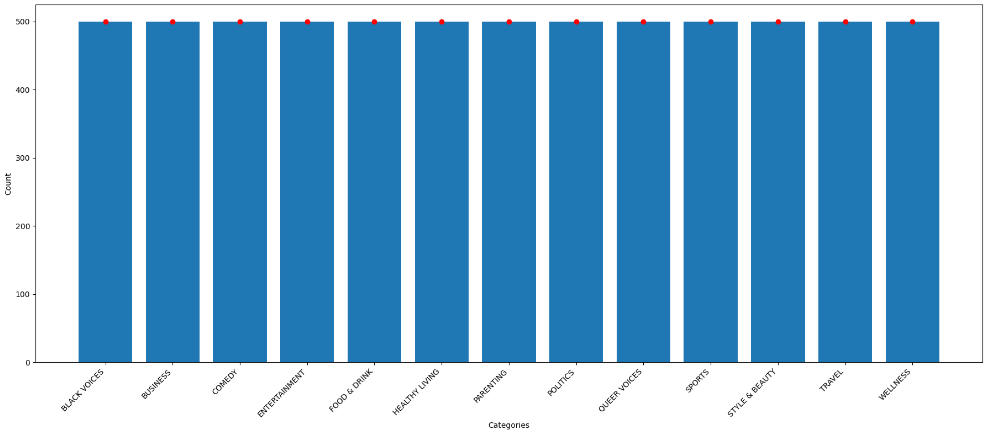


Figure 30 down sampled dataset visualisation

As the dataset have been balanced and thoroughly analysed NLP techniques are going to be performed in the dataset before vectorization and classification.

**Why do we need to balance our data?**

In NLP tasks, balancing the data before classification is crucial to developing strong, unbiased models that can accurately identify text input across all classes. To provide a balanced representation of classes in the dataset, methods including under sampling, oversampling, and class-weighted techniques can be used. In this case under sampling have been performed for balancing data.

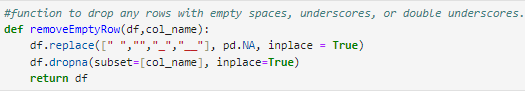
****

Figure 31 function to drop rows with empty space

****

Figure 32 checking data frame

This piece of code will remove all the empty rows from the dataset. In these case only one row have been removed from dataset which is good that we have all the data we expected except one row.

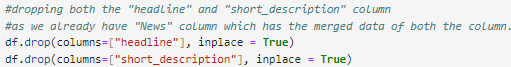


Figure 33 dropping headline & short description

‘Headline’ and ‘short\_description’ have been dropped as it have been concatenated into a new feature ‘News’ therefore it is no longer necessary for classification.

**Natural Language Processing (NLP)**

Natural Language Processing (NLP) is the branch of AI and computational linguistics that studies the interactions between computers and human (natural) languages. It includes the development of algorithms and models that allow computers to read, analyse, generate, and respond to human linguistic input in meaningful ways. It is commonly used for text classification/ categorization, Machine Translation, speech recognition etc.

These are the NLP techniques going to be performed on the data-

**Normalization**

Normalization in NLP refers to the process of transforming text data into a standardized, uniform format to make it easier to process and analyse.

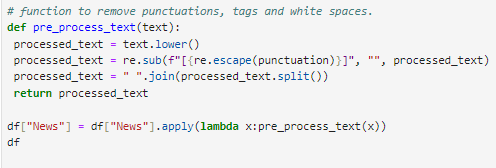


Figure 34 function for normalization

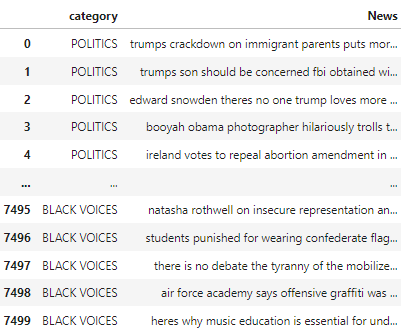


Figure 35 Output after Normalization

**Stemming**

Stemming is a text normalisation technique in natural language processing (NLP) that reduces words to their base or root form, also known as the word stem. The procedure entails removing suffixes or prefixes from words to obtain the root form, which may not always be a valid word in the language but represents the linguistic root.

****

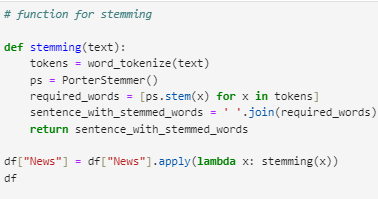


Figure 36 stemming function

Porter Stemmer algorithm have been used for the stemming and it is the most popular stemmer which is used around the world.

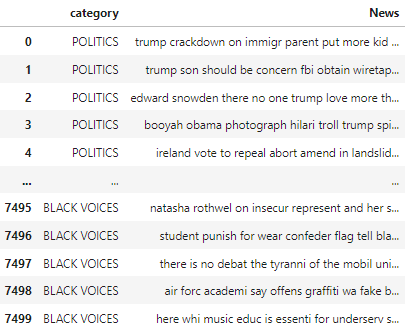


Figure 37 stemming output

**Lemmatization**

Lemmatization is the process of reducing words to their simplest or canonical form, known as the lemma. The primary goal of lemmatization is to group together a word's various inflected forms so that they can be analysed as a single unit.

****

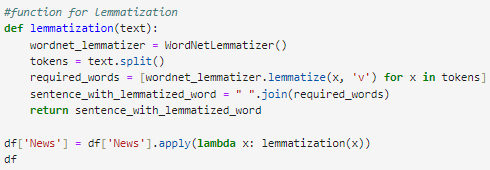
****

Figure 38 lemmatization function

WordNetLemmatizer use the help of WordNet to perform lemmatization by the help of its data in English language.

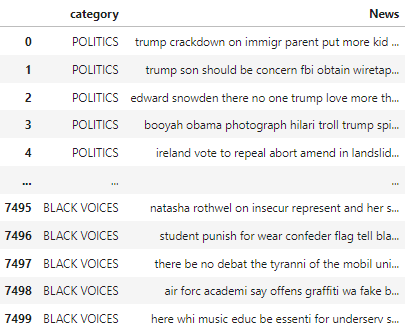


Figure 39 Lemmatization Output

**Stop word removal**

Stop word removal is a pre-processing step in natural language processing (NLP) in which common words considered uninformative or irrelevant to the task at hand are removed from the text.

****

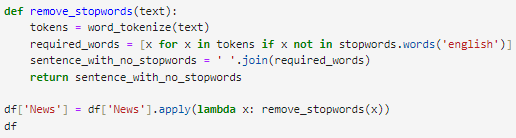
****

Figure 40 stop word function

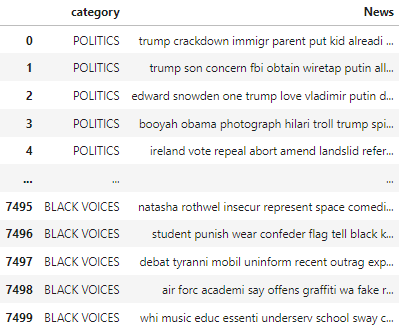


Figure 41 stop word removal output

All the essential NLP techniques have been performed on the text data other than vectorization. Now that the data have been thoroughly explored, analysed & pre-processed. The data is ready to be vectorised before classification.

# Vectorization of Data

Vectorization in Natural Language Processing (NLP) refers to the conversion of textual data into numerical vectors that machine learning algorithms can understand. It involves transforming text documents into numerical representations that can be used in tasks such as classification, clustering, and regression.

* **Hash Vectorization**

Hash Vectorization is one of the method used for vectorizing text data in Natural Language Processing (NLP). It is a type of vectorization that converts a collection of text documents into a matrix of token occurrences (counts).

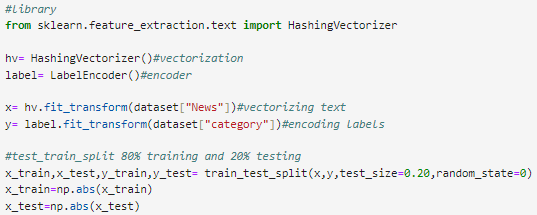


Figure 42 hash vectorization

* **Bag of Words (BoW)**

BoW (Bag-of-Words) is a common method used in natural language processing (NLP) for text vectorization. It represents text data as numerical vectors, where each dimension corresponds to a unique token (usually words) in the vocabulary, and the value of each dimension represents the frequency of the corresponding token in the text document.

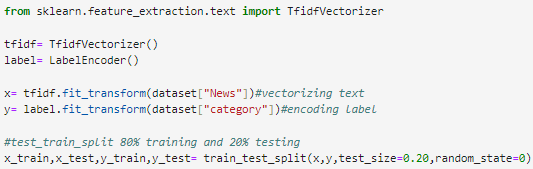


Figure 43 Bag of Words

* **TF-IDF**

TF-IDF (Term Frequency-Inverse Document Frequency) is a popular technique used in Natural Language Processing (NLP) to vectorize text data. It aims to capture the importance of words in documents by considering both the frequency of a term in a document (TF) and its rarity across the entire corpus (IDF).



Figure 44 TF-IDF

# Preparing Data

## Train-Test Split

The data have been split into training and testing set data using a single train-test split. The dataset have been split into training set and testing set with 20% testing set. This means 20% of the data have been reserved for testing and the remaining for training the model.

## Characteristics of Training set

The training data (x\_train and y\_train) are a subset of the original dataset (80% of it). It is used to train the machine learning model, enabling it to recognise patterns and relationships in the data.

## Characteristics of Testing set

The test data (x\_test and y\_test) is the remaining subset of the original dataset (20% of the total). It is used to evaluate the trained model's performance on previously unseen data. The model's predictions on the test data are compared to the actual labels to determine its accuracy and generalisation capability.

## Cross Validation

Cross Validation is a technique used to assess the performance of a model by splitting the dataset into multiple subset and training them individually and different manners. There are so many methods to perform cross validation. K-fold technique have been performed in this case with random forest classifier. K-folds were set to 5 which means 5 subset to be trained will be created and random state 42.

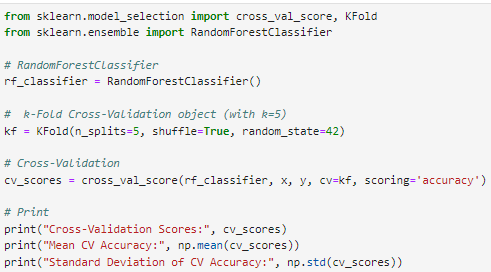


Figure Cross Validation

**Cross Validation Results-**

Hashing Vectorization



TF-IDF



Bag of Words



Highest accuracy obtained in cross validation is **0.70615385%**

Cross validation have only been performed on Random Forest Classification Model.

# Training and Evaluating Models

In this section, you should describe which model/s you have chosen for your learning task. Give a brief introduction to the model and provide justifications for your choice.

So far we have explored the dataset, performed different pre–processing on the data followed by the application various NLP techniques. Now we have decided which vectorization methods have to be applied moving on to the training of the different models. The goal is to experiment different vectorization approaches on different classification model to getter better accuracy after training. Author have experimented with 5 different model with different vectorization methods to get better results.

These are the Models used –

* Random Forest Classifier
* Multinomial NB
* Logistic Regression
* Support Vector Machines
* K-Nearest Neighbors

## 1. MODEL - Random Forest Classification

Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs their mode (classification) or mean prediction (regression). It is resistant to overfitting, can handle high-dimensional data, and performs well with minimal tuning. Random Forest can detect non-linear relationships in data and is capable of handling both numerical and categorical features, making it an excellent choice for text classification tasks.

# 1.1. Random Forest Classification - HASHING VECTORIZATION

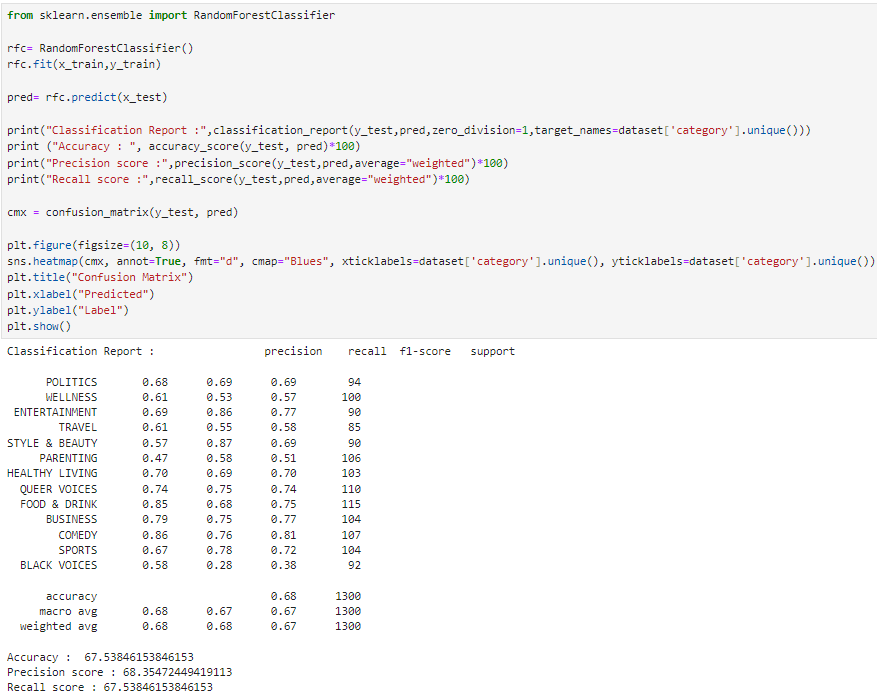


Figure Training Random Forest Classification - Hashing Vectorization

**Overall Metrics-**

Accuracy: **67.54%**

Precision: **68.36%**

Recall: **67.54%**

**Confusion Matrix –**

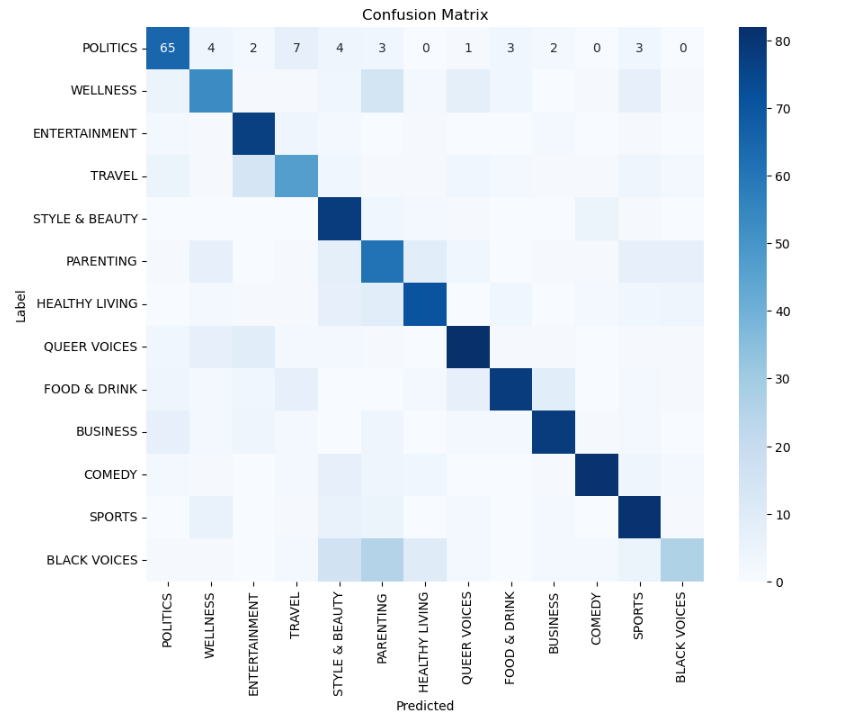


Figure Confusion Matrix

# 1.2. Random Forest Classification – TF-IDF

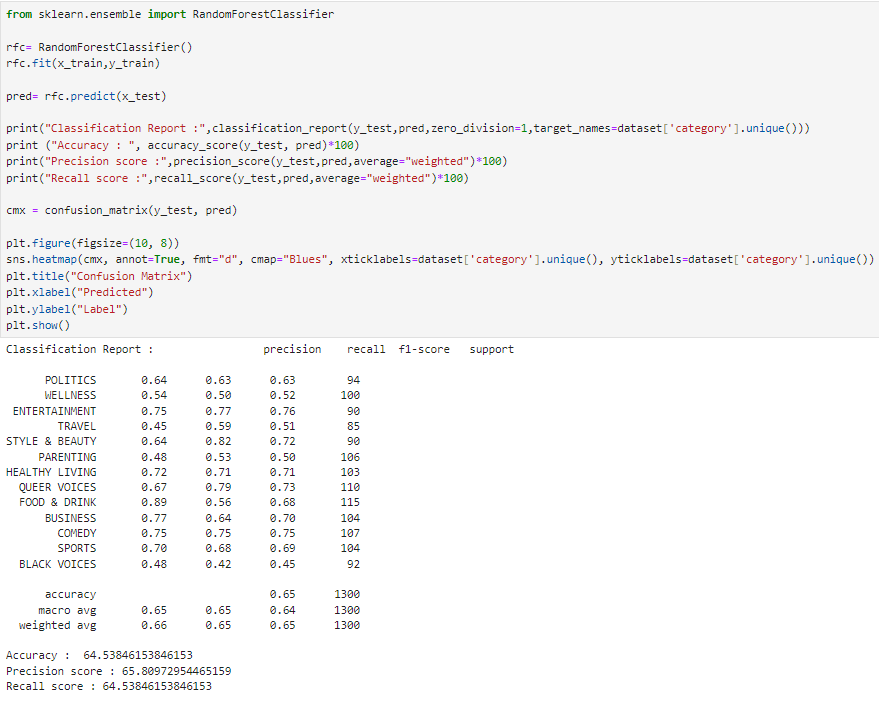


Figure Training Random Forest Classification - TF-IDF

**Overall Metrics-**

Accuracy: **64.54%**

Precision: **65.81%**

Recall: **64.54%**

**Confusion Matrix-**

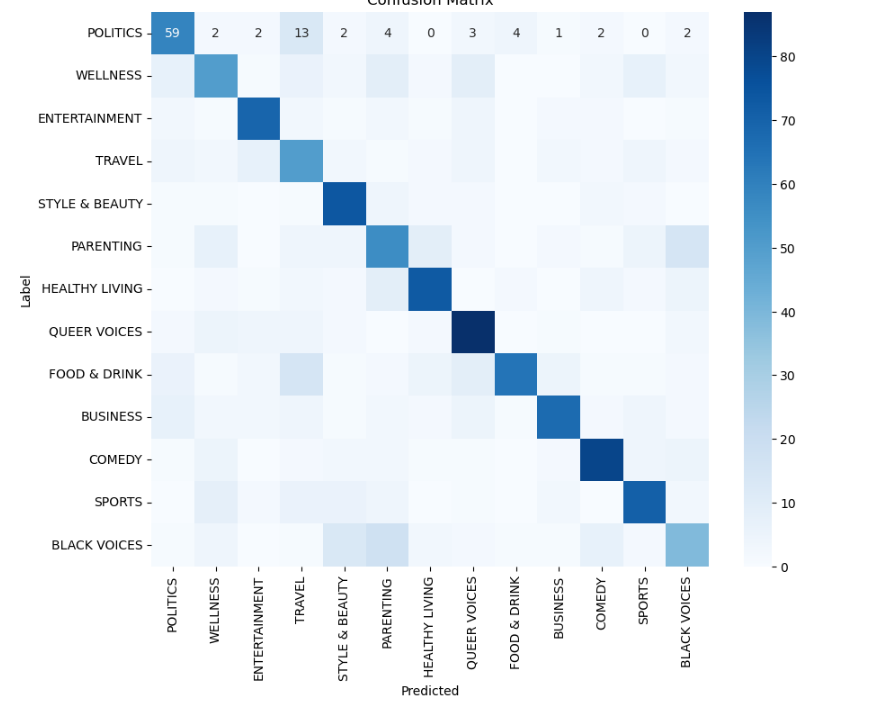


Figure Confusion Matrix

# 1.3. Random Forest Classification – BAG OF WORDS (BOW)



Figure Training Random Forest Classification - Bag of Words

**Overall Metrics-**

Accuracy: **63.39%**

Precision: **65.34%**

Recall: **63.38%**

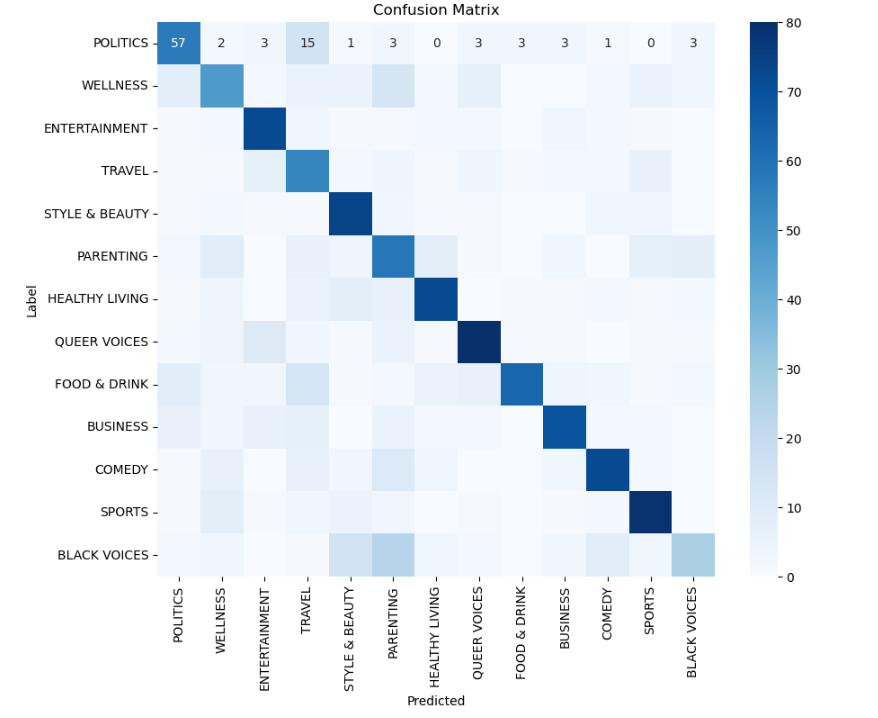


Figure Confusion Matrix

## 2. MODEL – Multinomial NB

Multinomial Naive Bayes is a classifier based on probabilities that uses Bayes' theorem and makes the assumption that features are independent. It is widely used in text classification tasks, particularly when dealing with features that represent word counts or term frequencies (such as TF-IDF vectorization). Naive Bayes is computationally efficient and performs well on high-dimensional data. It's ideal for text classification problems involving sparse feature matrices.

# 2.1. Multinomial NB - HASHING VECTORIZATION

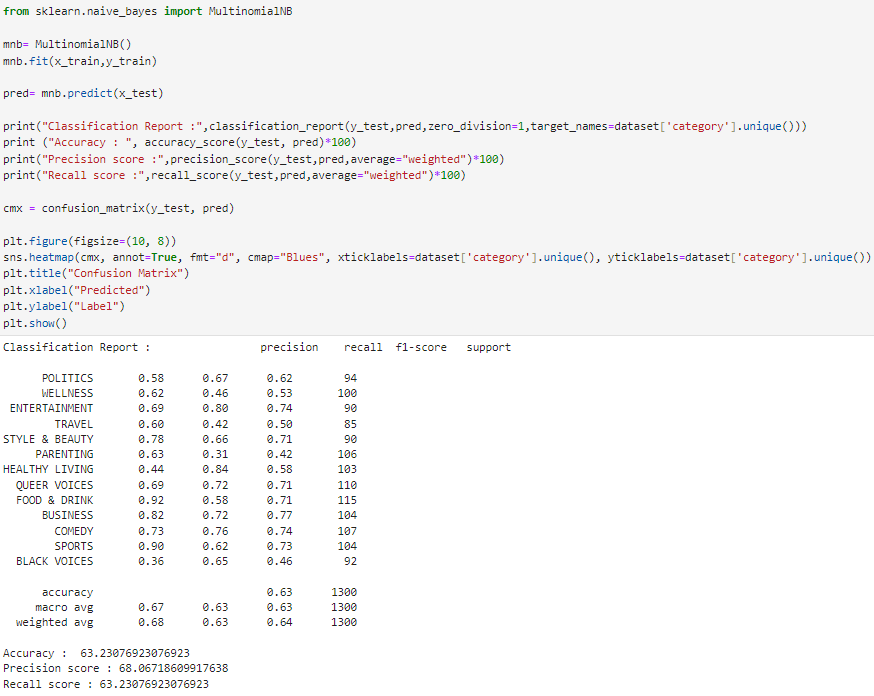


Figure Multinomial NB - Hashing vectorization

**Overall Metrics-**

Accuracy: **63.23%**

Precision: **68.07%**

Recall: **63.23%**

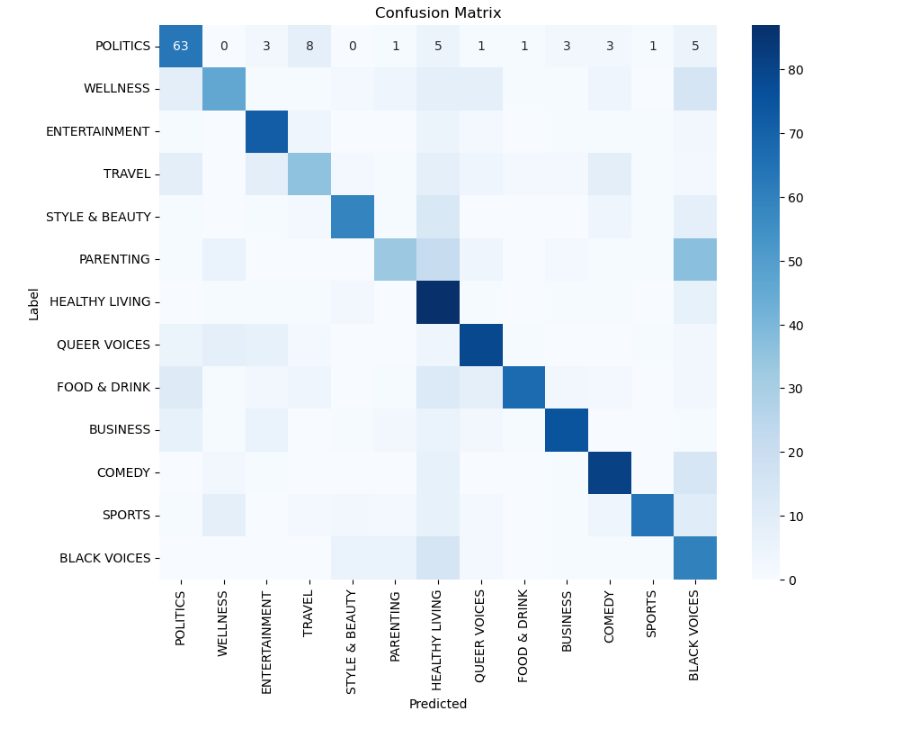


Figure Confusion Matrix

# 2.2. Multinomial NB - TF-IDF

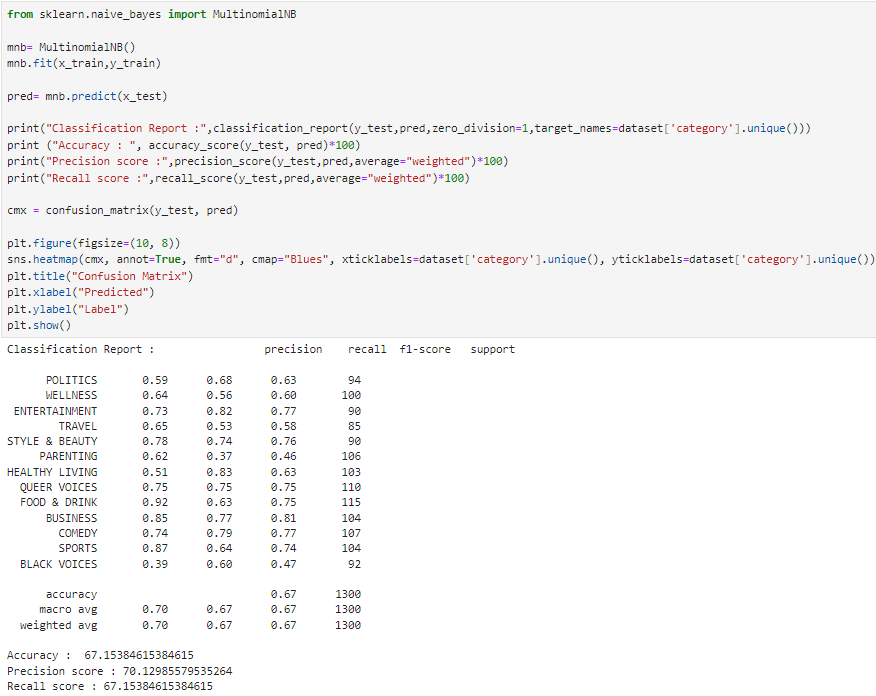


Figure Multinomial NB - TF-IDF

**Overall Metrics-**

Accuracy: **67.15%**

Precision: **70.13%**

Recall: **67.15%**

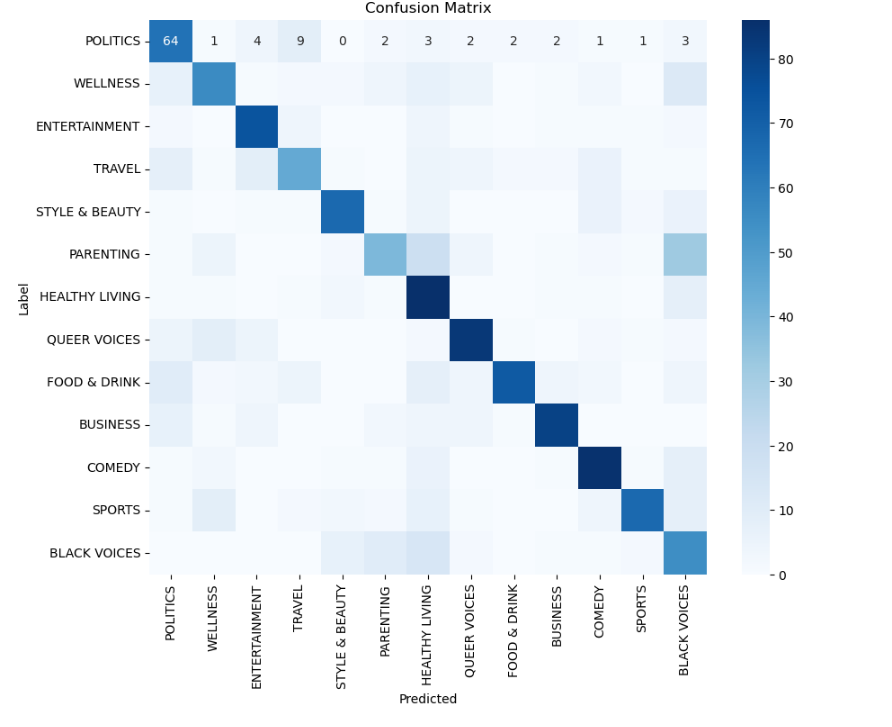


Figure Confusion Matrix

# 2.3. Multinomial NB - BAG OF WORDS (BOW)

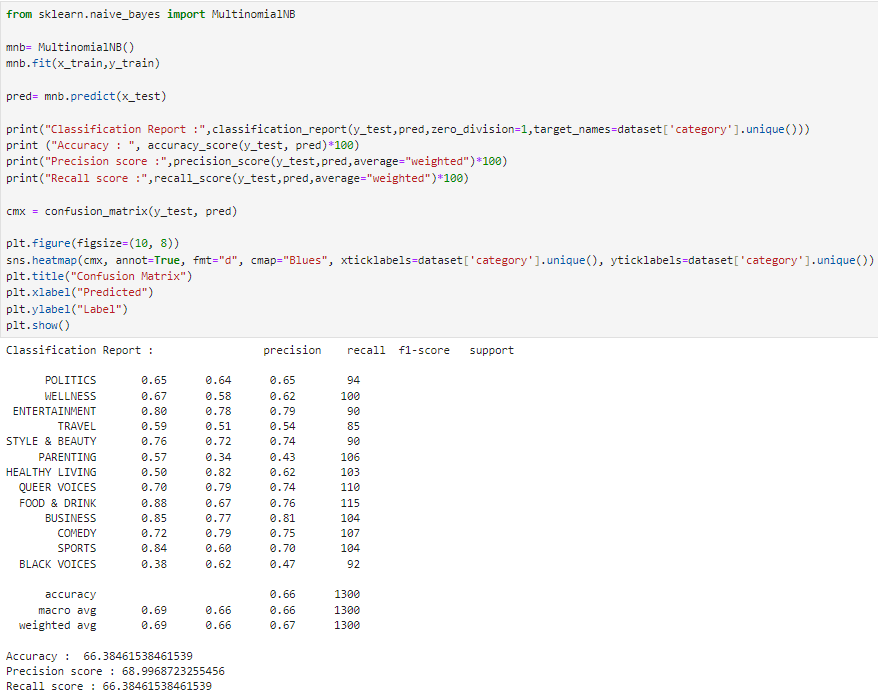


Figure Multinomial NB - BOW

**Overall Metrics-**

Accuracy: **66.38%**

Precision: **69.00%**

Recall: **66.38%**

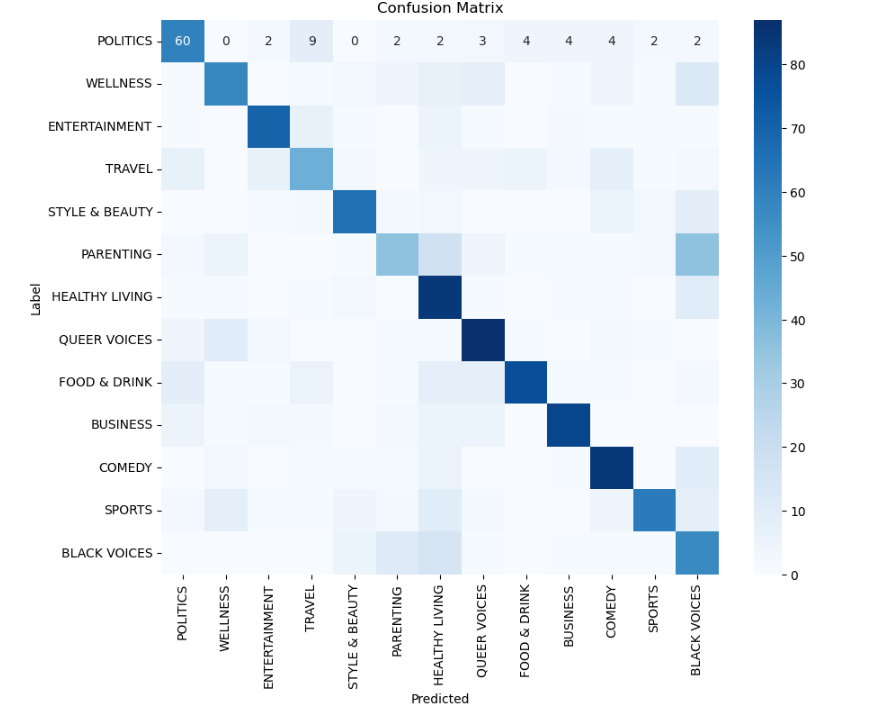


Figure Confusion Matrix

## 3. MODEL – Logistic Regression

Logistic Regression is a linear classification algorithm that uses a logistic function to calculate the probability of the target class given the input features. Despite its name, logistic regression is a classification algorithm that is widely used in binary and multi-class classification tasks. It is straightforward, understandable, and well-suited to tasks where the decision boundary is linear or can be approximated by a linear function.

# 3.1. Logistic Regression - HASHING VECTORIZATION

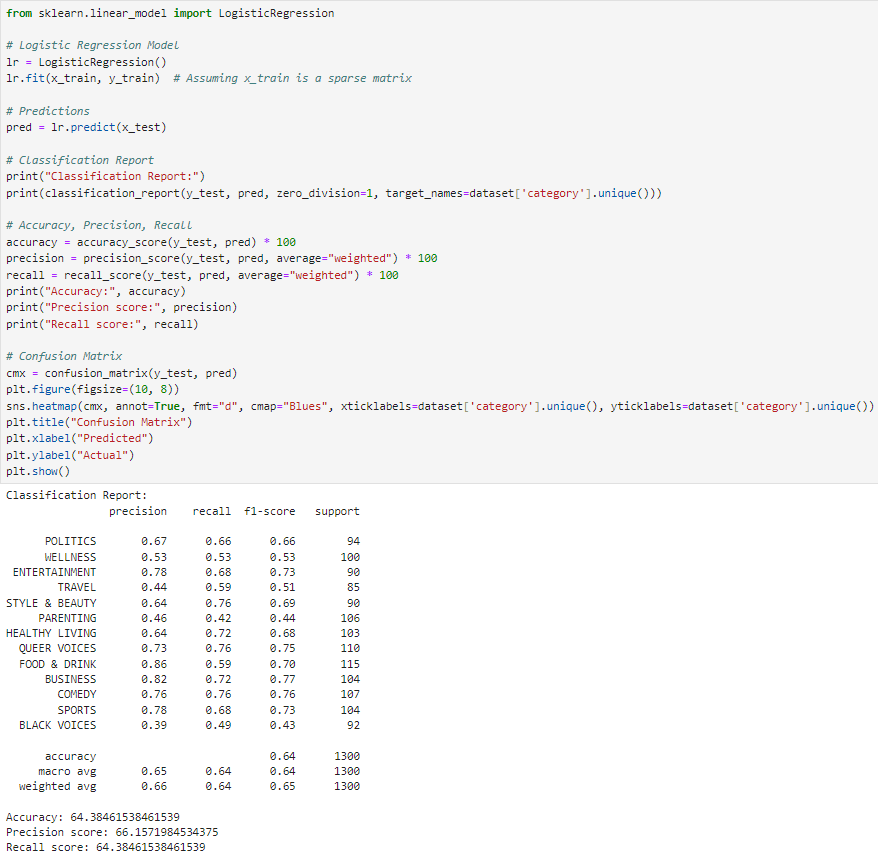


Figure Logistic Regression - Hashing vectorization

**Overall Metrics-**

Accuracy: **64.40%**

Precision: **66.16%**

Recall: **64.40%**

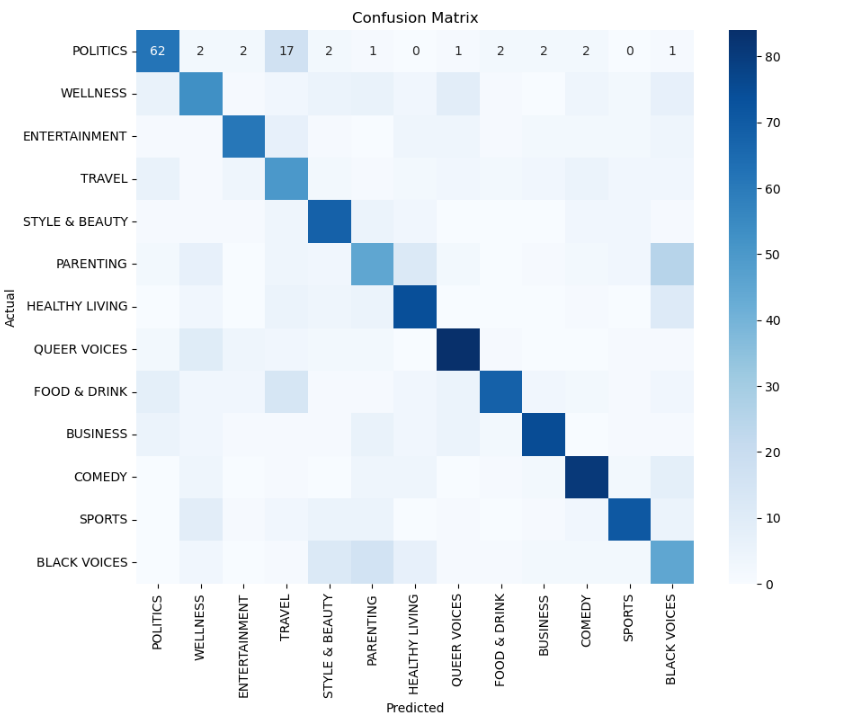


Figure Confusion matrix

# 3.2. Logistic Regression - TF-IDF

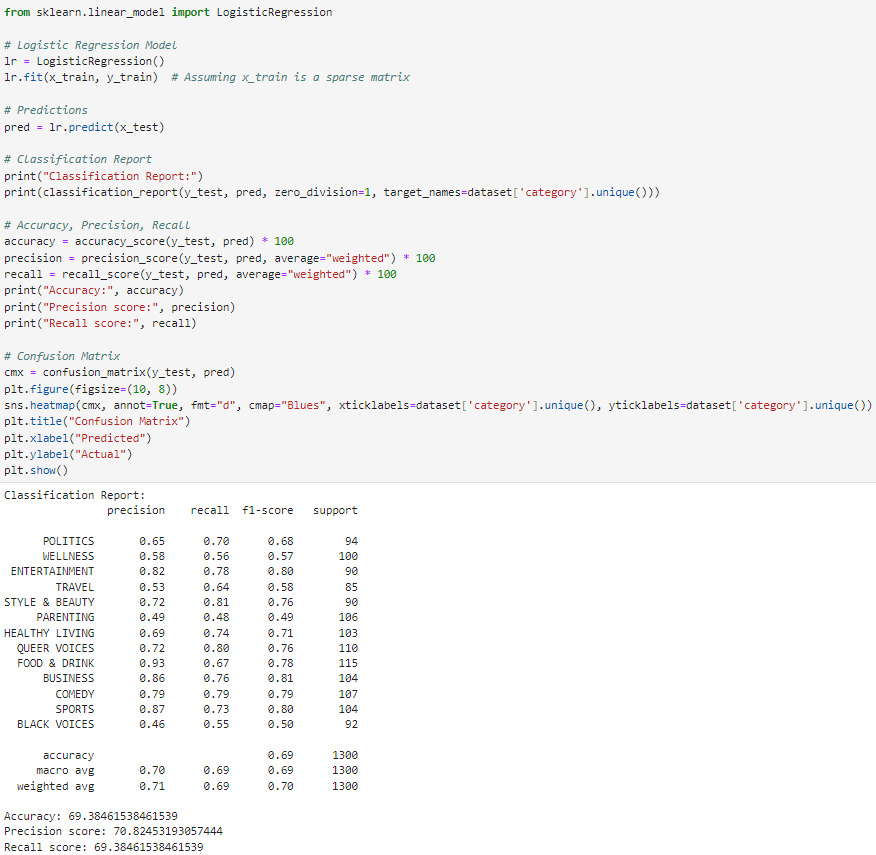


Figure Logistic Regression - TF-IDF

**Overall Metrics-**

Accuracy: **67.54%**

Precision: **68.36%**

Recall: **67.54%**

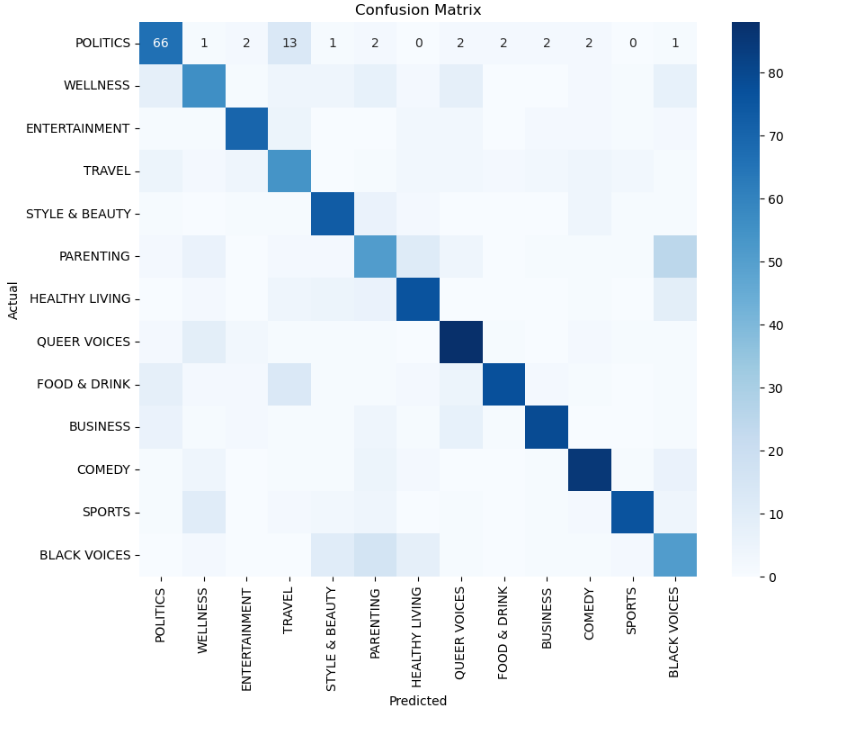


Figure Confusion Matrix

# 3.3. Logistic Regression - BAG OF WORDS

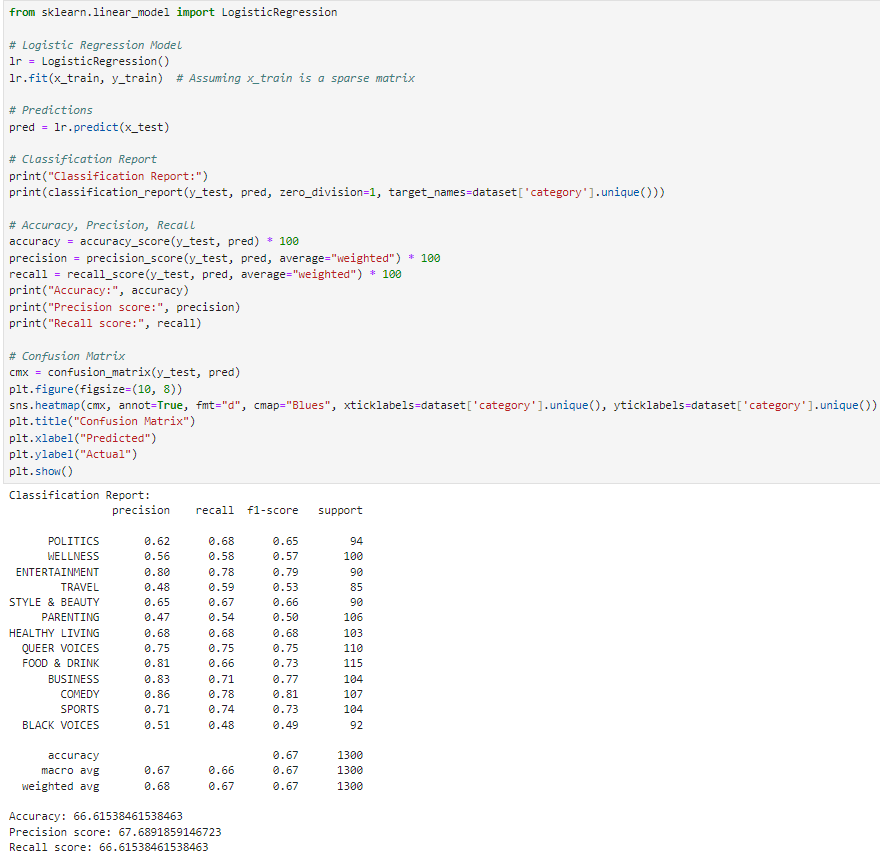


Figure Logistic Regression - BOW

**Overall Metrics-**

Accuracy: **66.62%**

Precision: **67.69%**

Recall: **66.62%**



Figure Confusion Matrix

## 4. MODEL – SVM

Support Vector Machines (SVM) is an effective supervised learning algorithm for classification and regression tasks. It determines the optimal hyperplane that separates the classes in the feature space by the greatest margin, thereby minimising the generalisation error. SVM can handle high-dimensional data and works well when there are more features than samples. It also supports a variety of kernel functions for dealing with nonlinear decision boundaries.

# 4.1. SVM - HASHING VECTORIZATION

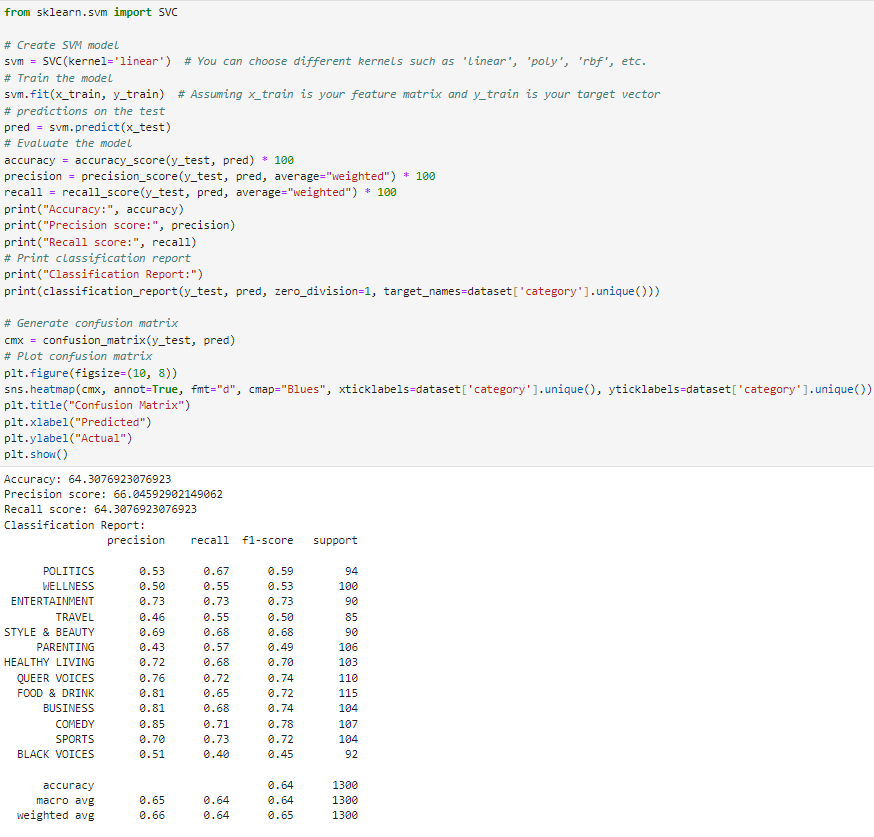


Figure SVM - Hashing Vectorization

**Overall Metrics-**

Accuracy: **64.31%**

Precision: **66.05%**

Recall: **64.31%**

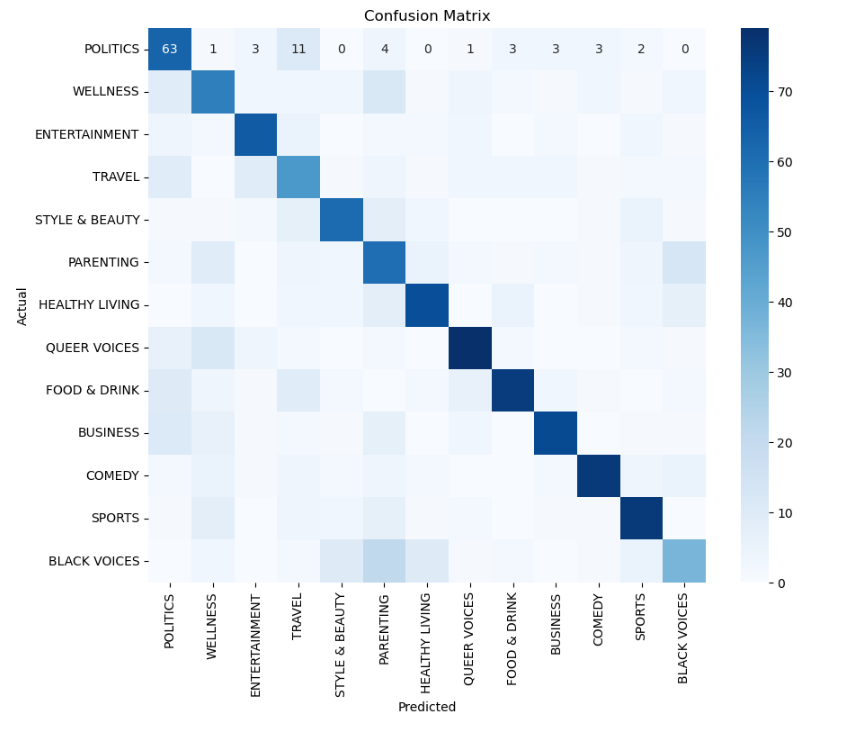


Figure Confusion Matrix

# 4.2. SVM - TF-IDF

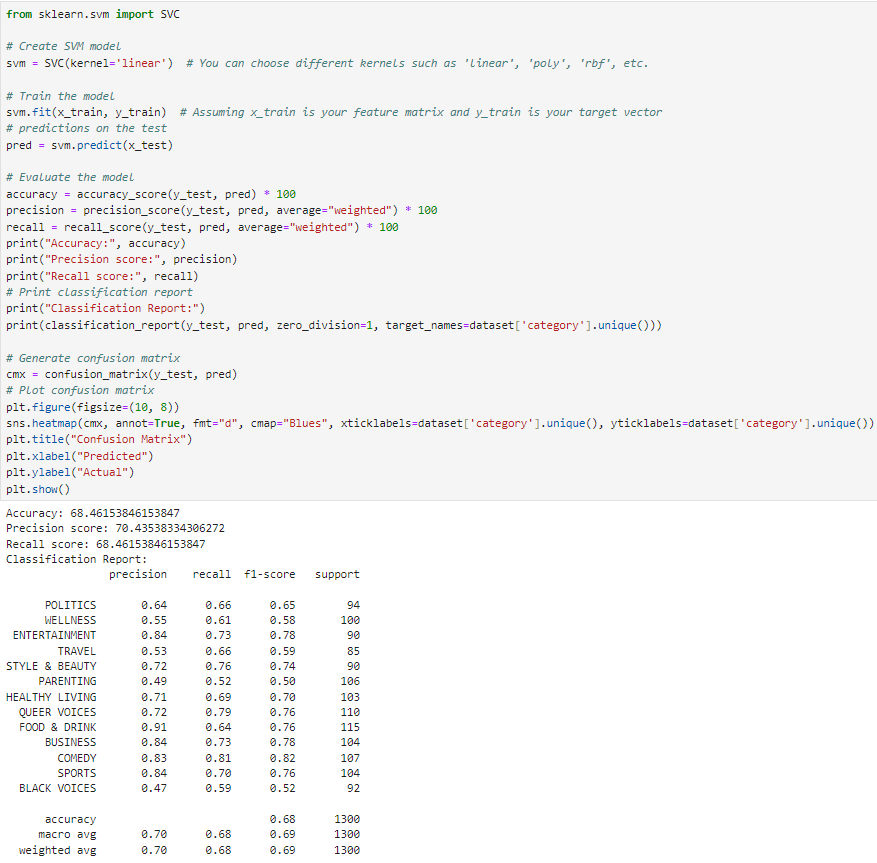


Figure SVM - TF-IDF

**Overall Metrics-**

Accuracy: **68.46%**

Precision: **70.44%**

Recall: **68.46%**

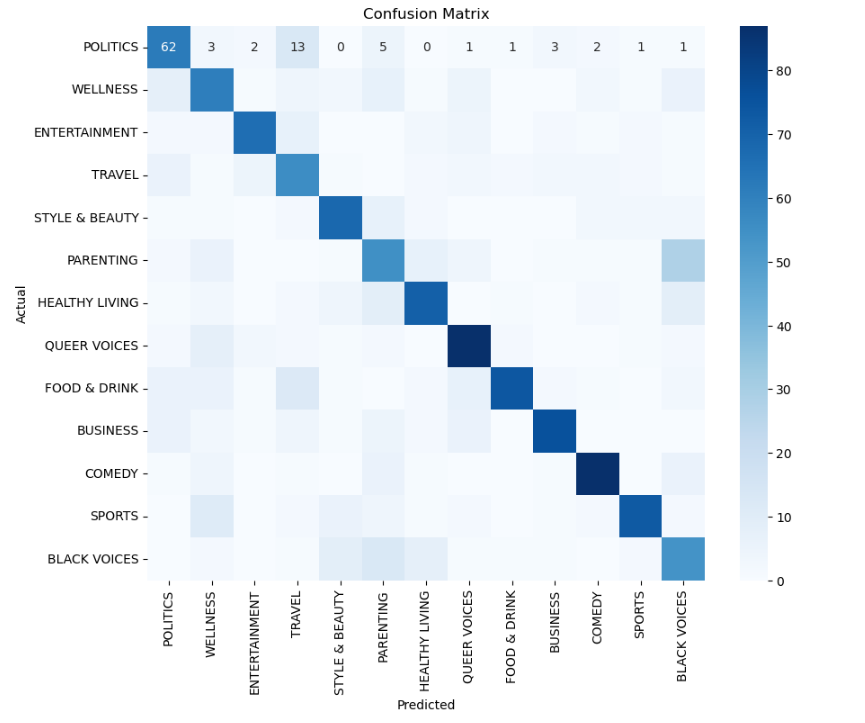


Figure Confusion Matrix

# 4.3. SVM - BAG OF WORDS

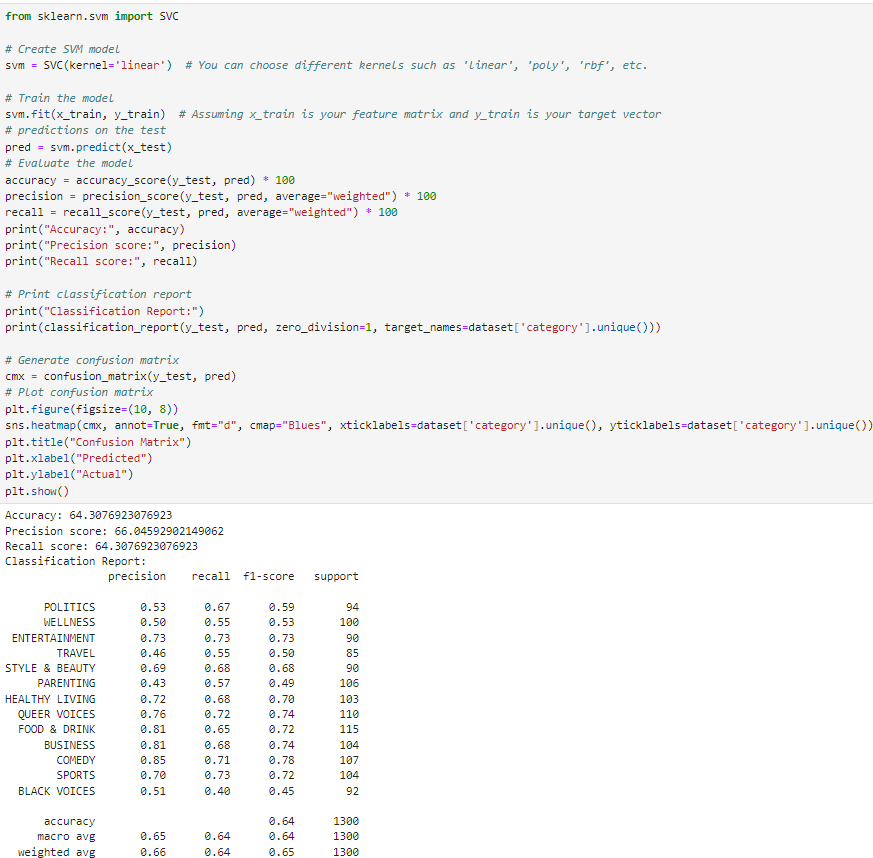


Figure SWM - BOW

**Overall Metrics-**

Accuracy: **64.31%**

Precision: **66.05%**

Recall: **64.31%**

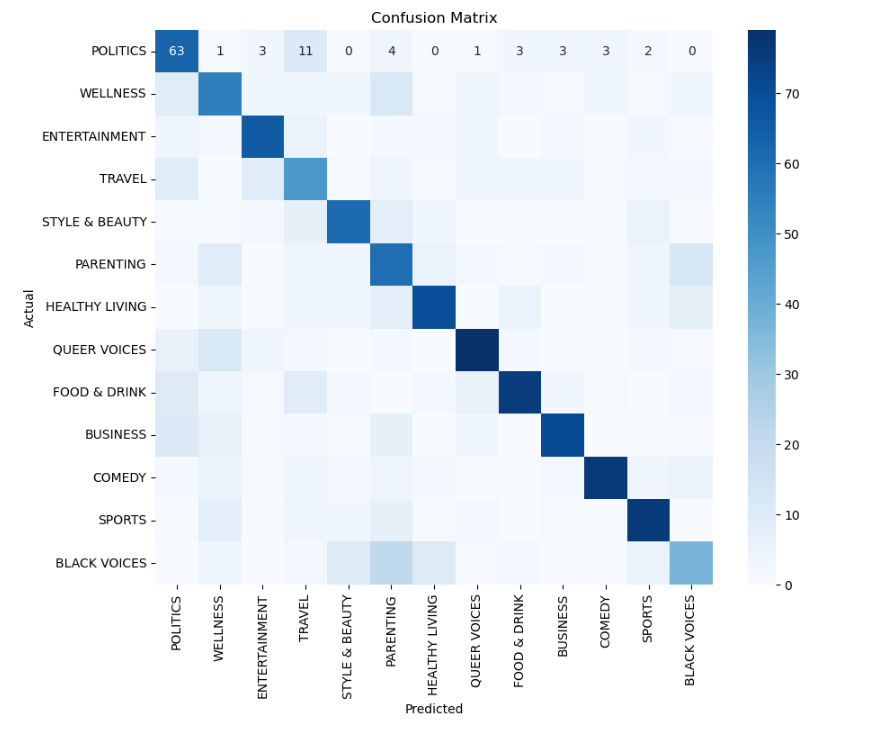


Figure Confusion Matrix

## 5. MODEL – KNN

k-Nearest Neighbors (kNN) is a simple and intuitive classification algorithm that assigns new data points based on the majority vote of their k nearest neighbours in the feature space. It does not require model training and is commonly used in scenarios where the decision boundary is irregular or difficult to define analytically. KNN can be computationally expensive, especially with large datasets, and it is sensitive to the selection of the number of neighbours (k) and the distance metric.

# 5.1. KNN – HASHING VECTORIZATION

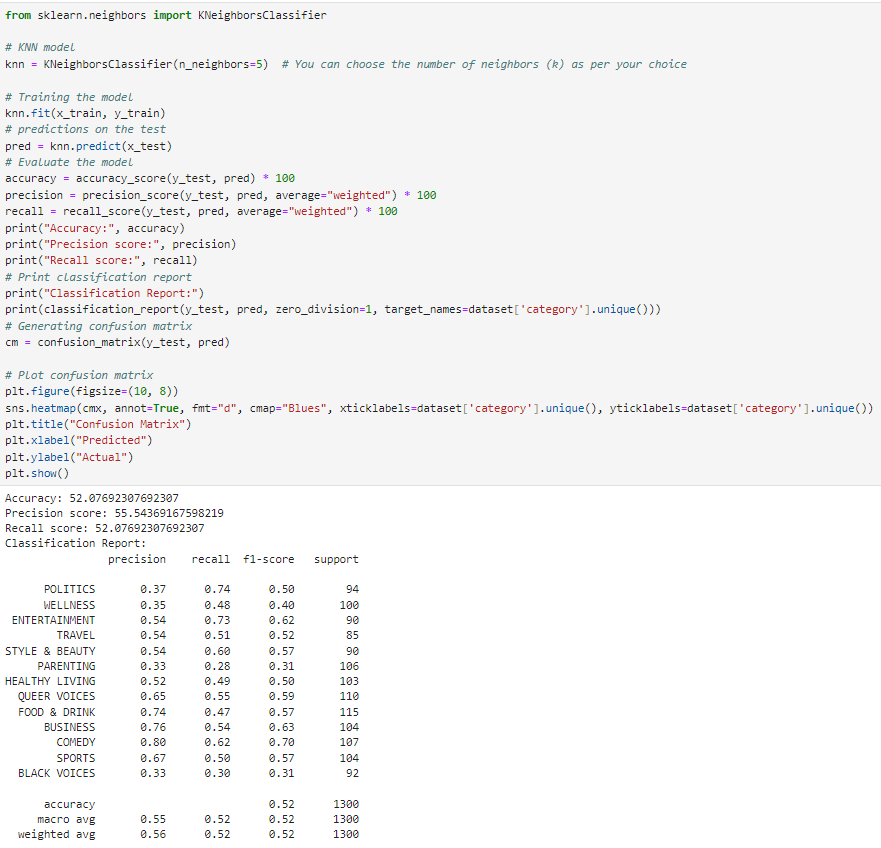


Figure KNN - Hashing Vectorization

**Overall Metrics-**

Accuracy: **52.08%**

Precision: **55.54%**

Recall: **52.08%**

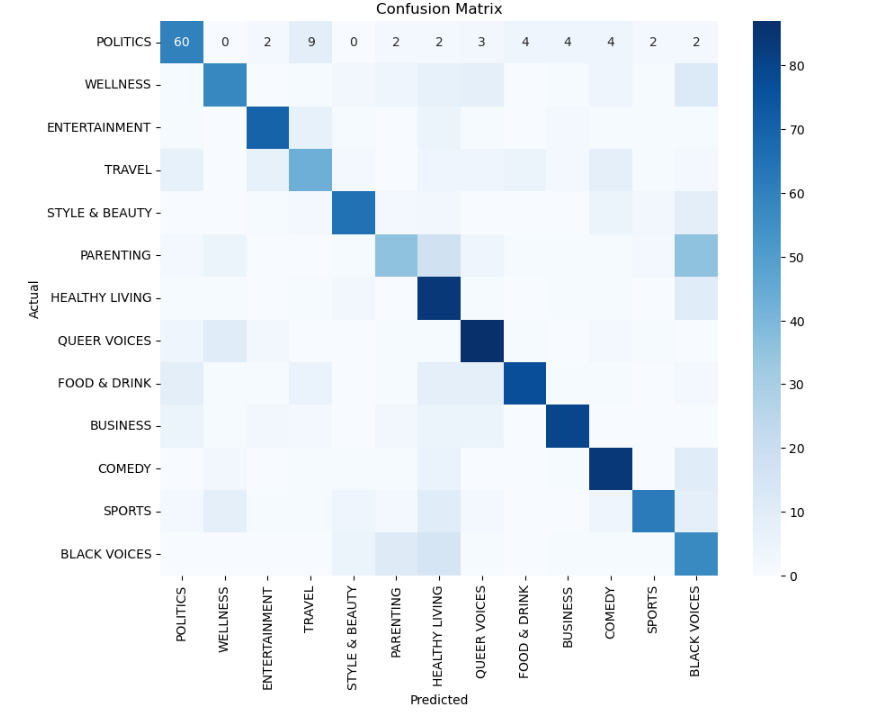


Figure Confusion Matrix

# 5.2. KNN – TF-IDF

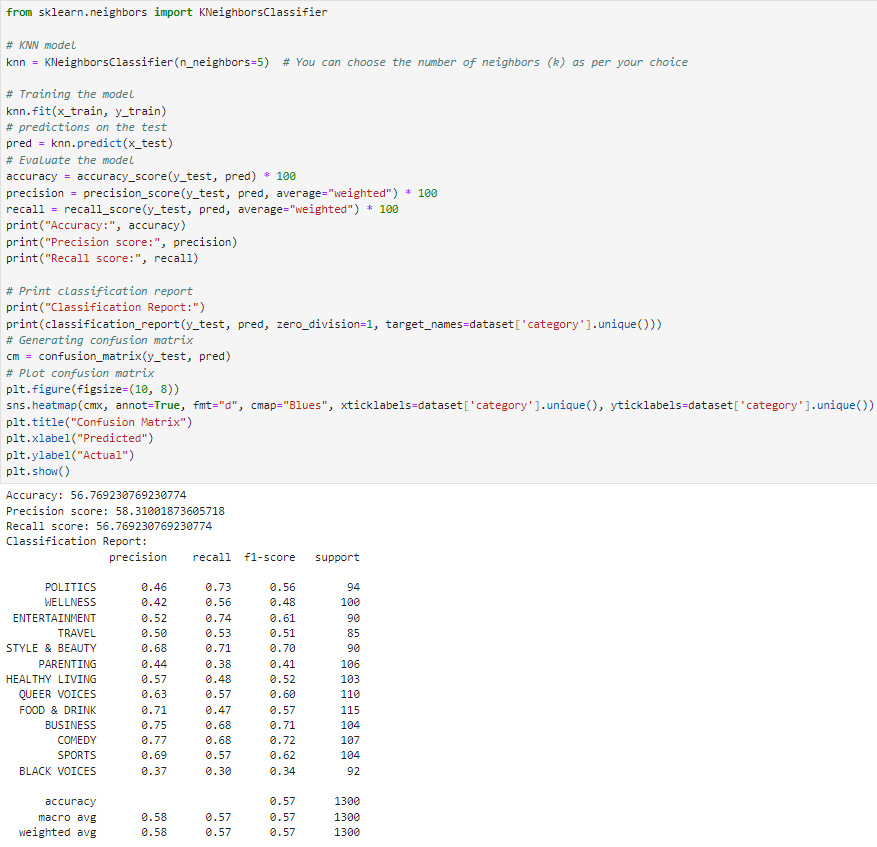


Figure KNN - TF-IDF

**Overall Metrics-**

Accuracy: **56.77%**

Precision: **58.31%**

Recall: **56.77%**

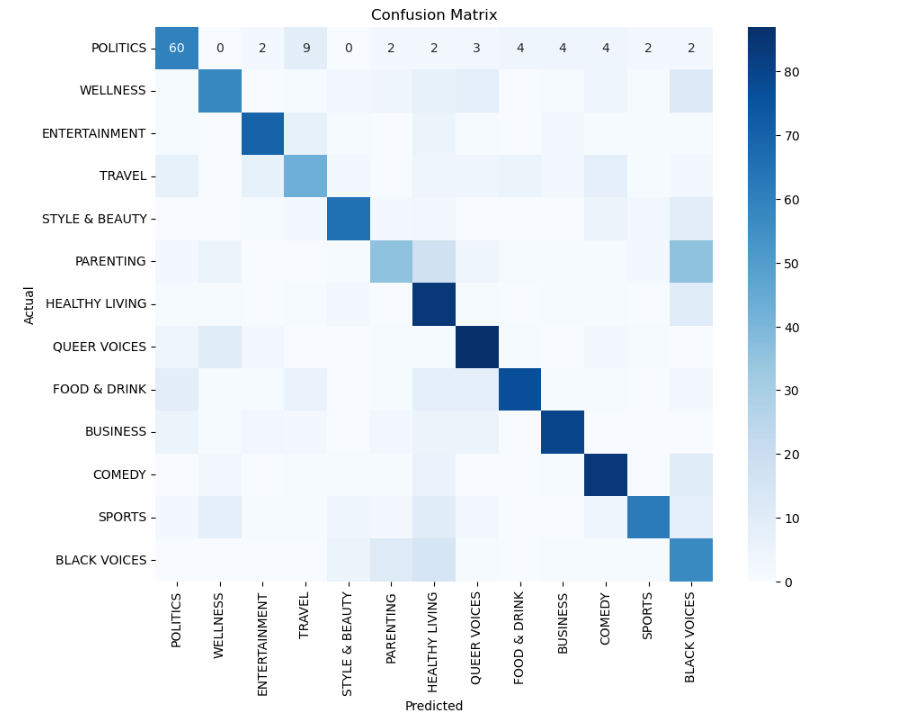


Figure Confusion Matrix

# 5.3. KNN – BAG OF WORDS

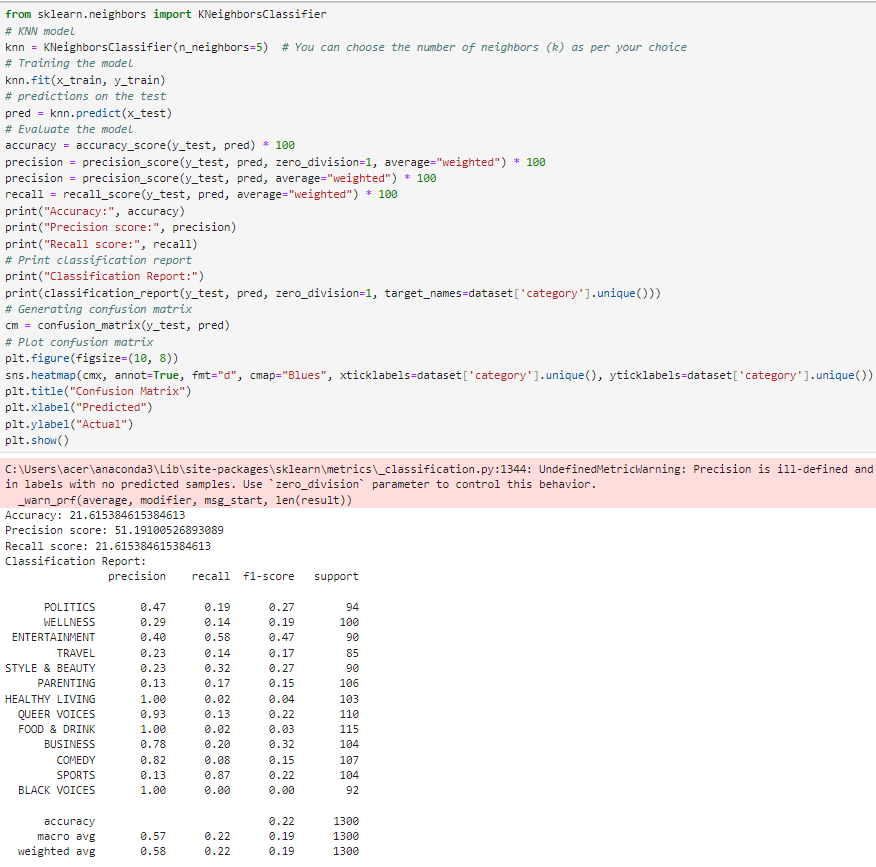


Figure KNN - BOW

**Overall Metrics-**

Accuracy: **21.62%**

Precision: **51.19%**

Recall: **21.62%**

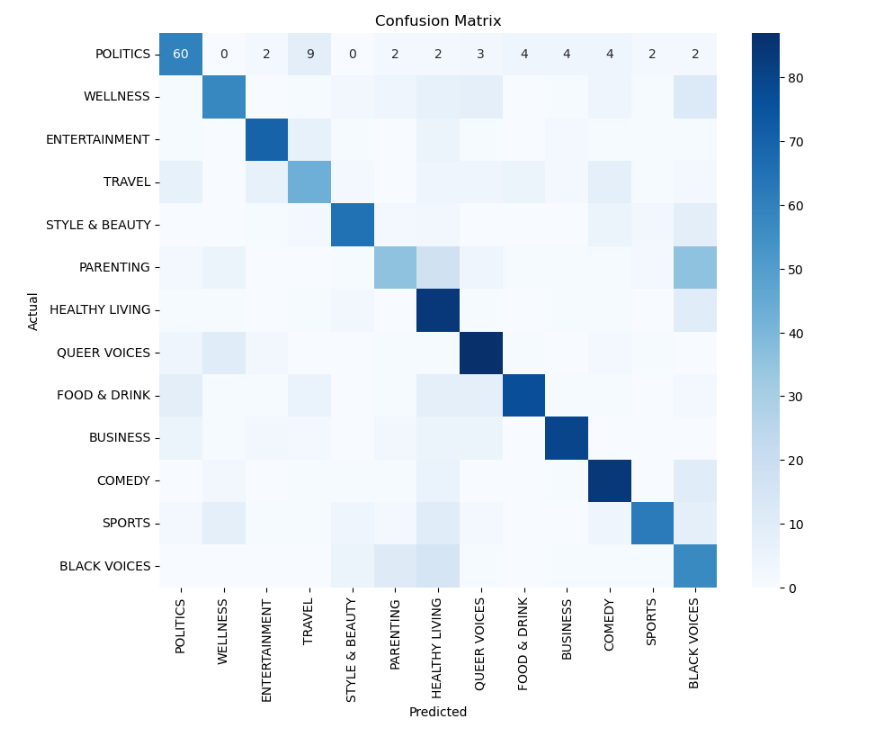


Figure Confusion Matrix

# Results/Evaluation Analysis

|  |  |  |  |
| --- | --- | --- | --- |
| MODEL ACCURACY % | | | |
|  | **HASHING VECTORIZATION** | **TF-IDF** | **BAG OF WORDS** |
| RANDOM FOREST CLASSIFICATION | **67.54%** | **64.54%** | **63.38%** |
| MULTINOMIAL NAÏVE BAYES | **63.23%** | **67.15%** | **66.38%** |
| LOGISTIC REGRESSION | **64.38%** | **69.38%** | **66.62%** |
| SUPPORT VECTOR MACHINES | **64.31%** | **68.46%** | **64.30%** |
| K-NEAREST NEIGHBOR | **52.08%** | **56.77%** | **21.61%** |

These are the result obtained from the training with different classification models using different vectorization techniques. As per the Accuracy % chart the Logistic Regression Model using TF-IDF was able to produce the best accuracy (69.38%) among all other models. These results are pretty good as we haven’t used much data for the training compared with the original dataset we have radically reduced the size of the data size due to the lack of computational efficiency. Better accuracy can be acquired if more data is used for the classification. Among the Models KNN Model using Bag of Words vectorization gave the lowest accuracy (21.61%). When considering the mean of all the accuracy of the models respective to their vectorization method TF-IDF is the most successful vectorization method to give better accuracy followed by Hashing Vectorization and Bag of Words. Whereas Logistic Regression is the classification model which gave the best accuracy when compared with other classification models

Evaluating the classification model by letting the model predict the category

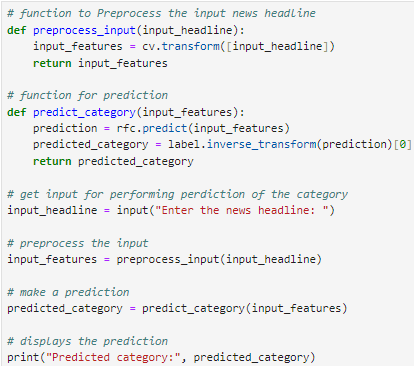


Figure code snippet for prediction of the category



Figure News Prediction 1



Figure News prediction 2



Figure News prediction 3



Figure News Prediction 4



Figure News prediction 5

These are couple of the predictions by the model which is really good for the amount of data that have been used for the training. Even though there was some misclassification on the 3rd and 5th news articles. Another thing to notice was the misunderstanding of the algorithm for the word apple which was meant as the brand but it have been labelled as food due to the same meaning of the word. Random Forest Classification model with TF-IDF was used to predict these news lines.

# Further Analysis and Methods

After all the processing and training I have decided to experiment the accuracy if we use more data and less categories. We are also trying hyper parameter tuning to determine if there is any improvement in the accuracy percentage. I have decided to consider only the first 3 categories ‘POLITICS’, ‘WELLNESS’, ‘ENTERTAINMENT’, all down sampled to 2000 articles, followed by performing the same NLP Techniques but chose not to proceed with stemming just to experiment if it make any difference. Saved the new data.

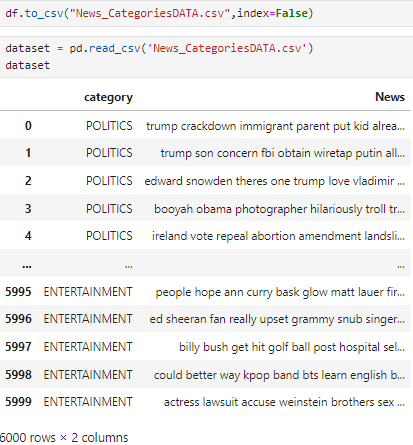


Figure New CSV

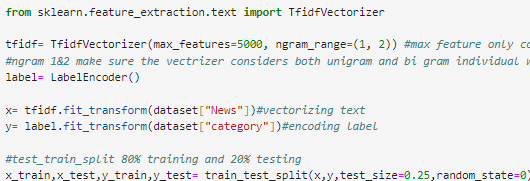
I chose to use TF-IDF as vectorizer as it was the best vectorizer among the three vectorization techniques used earlier. 

Figure TF-IDF

Max feature have been set to 5000 these makes sure that the vectorizer considers only 5000 features at a time. N-gram have been set 1 & 2 (uni-gram &bi-gram) these method makes sure that the vectorizer considers both individual and group of words from the dataset.

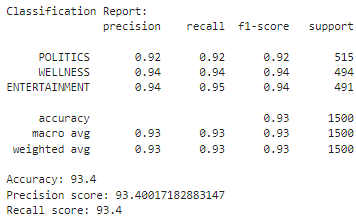
Tweaking the max feature from 100 to 5000 made a huge difference as the number kept increasing the models gave better accuracy. Accuracy was at the best when used max feature 5000.

Hyper tuning parameter grid search CV have been used along with logistic regression model the best score output was 0.93%

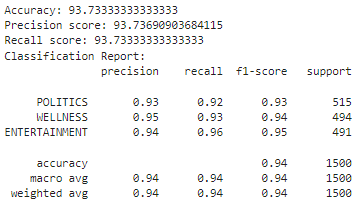


Figure Grid search CV

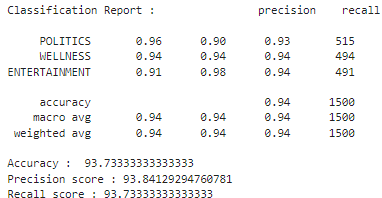
Multinomial Logistic Regression - 93.40%



Support Vector Machine – 93.73%



Multinomial Naïve Bayes – 93.73%



These were the accuracy % after increasing the data used for training and reducing the categories. We can see a clear hike in the accuracy percent from 60’s to 90’s 93.73% being the highest accuracy so far by Multinomial NB & SVM.

From this further analysis it is clear that exposing the models to more data would give us better accuracy and the usage of appropriate vectorization technique also plays a key role.

# Conclusion

To summarise, the Machine Learning module project, which focused on Natural Language Processing (NLP), Classification, and Vectorization, was both fascinating and benefiting. I've developed a better understanding of the complex processes involved in handling textual data, training models, and evaluating their performance through extensive research, experimentation, and analysis.

Throughout the project, I experimented with various pre-processing techniques to clean and prepare the dataset for analysis. Tokenization, stemming, lemmatization, and stop word removal were among the tasks assigned. Each step contributed significantly to improving data quality and model performance.

Vectorization proved to be a vital part of the project, as it required converting textual data into numerical representations that machine learning models could understand. I explored a variety of vectorization techniques, including Bag-of-Words, TF-IDF, and Hashing Vectorization. Each method had different benefits and contributed significantly to the models' performance.

I looked into various algorithms and methodologies for effectively classifying the dataset. This included traditional machine learning classification algorithms, such as KNN, SVM, Multinomial NB, Logistic Regression and Random Forest.

The models were diligently evaluated using metrics such as accuracy, precision, recall, and the F1-score. This allowed for a thorough evaluation of each model's performance and facilitated comparisons of various approaches. Furthermore, techniques such as cross-validation were used to improve model performance and reliability.

As a further experimentation with the data I have decided to increase the data used with only 3 categories it gave excellent accuracy when compared to the previous method, as it have been exposed to more data for the model to learn. Also the hyper parameter tuning using grid search played critical role in giving an insight about the model and their accuracy.

In the end, the project has not only increased my knowledge and understanding of machine learning techniques, but it has also provided me with practical skills that will be useful in real-world applications. Moving forward, I am excited to continue researching and experiment with different Deep Learning Models such as LSTM hoping to get an even better result than the one I have acquired.

# References

[Visualization](https://plotly.com/) - plotly.com

[Natural Language Processing NLP](https://www.datacamp.com/tutorial/tutorial-natural-language-processing?utm_source=google&utm_medium=paid_search&utm_campaignid=19589720821&utm_adgroupid=157156374951&utm_device=c&utm_keyword=&utm_matchtype=&utm_network=s&utm_adpostion=&utm_creative=684592139651&utm_targetid=dsa-2218886984380&utm_loc_interest_ms=&utm_loc_physical_ms=9044911&utm_content=&utm_campaign=230119_1-sea~dsa~tofu_2-b2c_3-row-p1_4-prc_5-na_6-na_7-le_8-pdsh-go_9-na_10-na_11-na&gad_source=5&gclid=EAIaIQobChMItaKY3ZOphQMVK5JoCR1a1wN8EAAYAiAAEgI7r_D_BwE) – datacamp.com

[Logistic regression](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html) – Sci-kit

[Vectorization](https://machinelearningcompass.com/machine_learning_math/vectorization/) – Machine Learning Compass

[News classification using ML](https://www.kaggle.com/code/hectortiendrebeogo/news-classification-using-ml-73-6#Now,-let's-create-some-function-to-perform-algortihm-on-data) - Kaggle

[Multiclass text classification](https://towardsdatascience.com/multi-class-text-classification-model-comparison-and-selection-5eb066197568) – towards datascience

[Comparing Classification model](https://www.kaggle.com/code/kushalnigam/comparing-classification-models) - Kaggle