MACHINE LEARNING TEXT ANALYZER - TEXT CLASSIFICATION USING SUPERVISED AND UN-SUPERVISED ALGORITHMS

A Thesis submitted to the

Department of Computer Science and Engineering

African University of Science and Technology

In partial fulfillment of the Requirements for the Degree of

Master of Science in Computer Science

By

Ezeaneche, Ifeoma Amaranna

June 2019.

CERTIFICATION

This is to certify that the thesis titled “Machine Learning Text Analyzer - Text Classification Using Supervised and Un-Supervised Algorithms” submitted to the department of Computer Science and Engineering, African University of Science and Technology Abuja, Nigeria, for the award of Master’s Degree is a record of original research carried out by Ezeaneche Ifeoma Amaranna in the department of Computer Science and Engineering.

African University of Science and Technology [AUST]

Knowledge is Freedom

APPROVAL BY

Supervisor

Surname: Ekpe

First name: Okorafor

Signature: [illegible] 21st June 2019

The Head of Department

Surname: DAVID

First name: Amos

Signature: [illegible]

COPYRIGHT

ABSTRACT

Text analysis is a branch of data mining that deals with text documents. This project brings to light the classification of texts into their various categories. The structured and unstructured data seems to on a high rise in this era. Thus, to be able to classify this data is important. Classification however starts from collection, preprocessing, and feature extraction. There are several techniques that can be used for text classification, but machine learning algorithms will be employed in this project. Because of the advent of Natural Language Processing, we will be able to see the need for feature extraction and selection.

In this research, we will be able to see how the computer intelligently classifies text into their various categories. Emphasis will be on English language word document.

Key words: Matching Algorithms (ML), text classification, MNB, KNN, SVM, LR.

DEDICATION

I dedicate this work to God, to my parents, Sir and Lady Ben Ezeaneche, to my siblings and to myself.

ACKNOWLEDGEMENT

With much appreciation to God Almighty, I will like to extend my heart felt gratitude for giving me the grace to complete this project.

Furthermore, I will like to appreciate my supervisor, Dr. Ekpe Okorafor who has taken his time to be diligent in his work. The guidance, mentorship and support are appreciated in so many ways.

In addition, I thank the staff and faculties of Computer science who have impacted knowledge in me.

More so, I will like to appreciate the support of my family, Sir and Lady Ben Ezeaneche, Obi, Nne, Elo and Chuchu for their unending love and support. Thank you for always being there for me while growing up, and for teaching me.

I will also like to thank my classmates for their camaraderie. Thank you.

Contents

LIST OF FIGURES

LIST OF TABLES

CHAPTER1

1.1 Introduction

Text analysis is a field that has seen been growing rapidly over the years. The idea of text analysis came into being around the year 1950 ("Text Analytics: A Primer | GreenBook,"2017). Its main objective is to enable a descriptive view of structures and contents of a text document.

Text analysis is the act of understanding or deriving important information contained within a document or text. While structured data is generally being managed using a database system, text data is typically managed using a search engine owing to the fact that unstructured data is involved. The percentage of unstructured data generated has been increasing rapidly as the years go by. The growth rate is about 55% to 65% each year as statistical records show ("Structured vs. Unstructured Data," 2015). Analyzing text based on sentimental views is of utmost importance in this era. To understand or predict the emotional balance or secret messages in a subjective context helps the data analysts in doing the desired job.

Data analysis helps to obtain reasonable facts that can be used as a company's marketing strategy. Responses from individuals can be used to determine when products are to be produced in bulk. Using sentiment analysis can help in determining positive or negative views about a company (Saranya & Jayanthy, 2018), confidential and non-confidential messages to be seen by the public can also be determined through its use.

Managing huge data set is quite difficult to handle, hence, the reason text classification comes into play. Text classification is the act of classifying or arranging a large amount of data generated into different or pre-defined categories as required. Without these classifications, it would be difficult to accumulate data. Proper arrangement of these data makes work easy. Despite making work easy, data classification requires a lot of work. Hence the reason for the introduction of machine learning concept.

The purpose of using machine learning is to be able to develop a suitable algorithm that can be used to ensure intelligent data classification so as to augment system performance using the trained dataset. In other words, machine learning helps to solve intelligent text classification problem. There are certain classification steps to be followed in order to do this; they are as follows:

i. **Data Collection:** This involves accumulating data needed for the experiment.
ii. **Text Preprocessing:** Preparing the raw data for another process. It includes removing unwanted spacing, capitalization, etc.
iii. **Feature Extraction or Selection:** This involves selecting and reducing the number of randomized variables within the text.
iv. **Text Representation:** Applying supervised and unsupervised learning algorithms on the dataset.
v. **Text Classification and Processing**: Grouping of text into categories and using the training and test datasets to arrive at the result

1.2 Natural Language Processing (NLP)

This is a significant research area that deals with ways of enabling computers to understand and interpret human language. Ordinarily, computers do not really understand human language. NLP is based on deep learning which is a research area in machine learning (Nene, 2017). It has been used in various field of studies such as robotics, electrical, and computer engineering, etc. (Dr. S. Vijayarani, Ms. J. Ilamathi, 2018). It comprises different algorithms that can be used to collect data on how human beings understand and use languages and for transforming the unstructured data for the computer understands. Although, acting as the middleman between computers and humans, it can be applied in various sentiment analysis applications in areas such as speech recognition, stemming, artificial intelligence, text summarization, etc. It helps to dissolve the structure and obtain essential information for the computer.

When text or data is provided, the computer makes use of some algorithms that can be used to mine information from each sentence in the dataset and store the important data ("An easy introduction to Natural Language Processing," 2015). Sometimes, the computer may fail to properly understand the meaning of a sentence, thus, leading to obscure results like the translation of words from English to the Russian language that occurred in the 1950s. However, it does not disprove the fact that natural language is hard to discern by computers.

1.3 Objectives of the Project

The aim of this project is to be able to classify text according to its topics using supervised machine learning algorithms in order to classify data by employing machine learning algorithms. It has been pointed out that applying machine learning algorithms can help in achieving the desired result. Further, we also aim to compare the results obtained using the proposed model with the results obtained using other related models in terms of accuracy.

1.4 Problem Statement

While large datasets have been made available over the internet, classification of these datasets is becoming increasingly important. Further, several projects that have been carried out in the past could not come up with a way to effectively perform data categorization despite the different types of features that were used (Saranya & Jayanthy, 2018). N-gram features (unigram and bigram) have often been used. In this project, we will focus on the use of 4-gram feature in performing data classification of a text analyzer whilst keeping in mind the end goal of achieving maximum accuracy.

1.5 Limitations of the Study

This research is limited to only the implementation of machine learning algorithms with huge datasets in a big data platform. The platform intended to be used is the Hortonworks platform, however, due to space and unreliable power supply challenges, it was not used.

1.6 Chapter Organization

This project entails the steps and results from text classification and analysis. It comprises five (5) chapters. Each of the chapters discusses distinct topics. Below is a brief description of the chapters:

i. Chapter 1 discusses an introduction to text analysis, sentiment analysis, natural language processing, aims and the limitations of the project.
ii. Chapter 2 focuses on machine learning concepts to be used, the classification of datasets and its comparison with previous works/projects
iii. Chapter 3 presents the necessary materials that will be used to accomplish this project, the methodology, its setup and demonstration of how the classification will be implemented.
iv. Chapter 4 discusses the implementation of the work and the results obtained.
v. Chapter 5 presents the conclusion of the project.

CHAPTER 2

2.1 Introduction to Machine Learning

Machine learning (ML) is a container of algorithms that can be used to predict and classify computer applications/software in an intelligent manner. It can be used to statistically determine through analysis techniques, ways to predict the outcome of data whilst using the input data that was trained to make the necessary predictions. The idea of computer task being automated has been as a result of ML. ML can be defined in so many ways.

According to (Nilsson, 2005), ML refers to the changes in systems that perform tasks associated with artificial intelligence (AI). Such tasks include recognition, diagnosis, planning, robot control, and prediction.

ML can also be defined as a branch of artificial intelligence that allows computer systems to learn directly from examples, data, and experience (The Royal Society, 2017).

Further, (Kose, Harankhede, Shukla, &Mohod, 2018) defined ML as a band of algorithms that can be used to express classification rules or classifiers automatically using example documents**.**

ML makes it possible for computers to be able to understand the human language and to learn from a given set of data. ML algorithm uses different computational methods for learning.

There are three (3) types of techniques used in machine learning which are

i. Supervised learning

ii. Unsupervised learning

iii. Semi-supervised learning

2.1.1 Supervised Learning

The input and output datasets are used in this technique to train a model so that the output prediction of a dataset can be known using the trained model. The aim of supervised learning is to be able to build a system that can predict outputs based on the results of the input data that the ML model was trained with. The uncertainty of the future is being predicted. The input is usually known. The algorithm takes the input and trains it. The response that is being generated is used to test the remaining dataset for possible predictions. It applies what has been learnt from the past data to newly presented data for prediction (“Supervised and Unsupervised Machine Learning Algorithms,”2017. Its main predictive models are the classification and regression models. This learning algorithm can make matches with the supposed output and the output gotten from the training, find errors in the results, and can be used to modify the system/model into the desired form. Learning continues until a level of satisfactory outcome for the training data is achieved. Most times, supervised learning techniques are used in machine learning systems. The input data is tagged X while the output data is tagged Y. An algorithm can be used to map the input data to the output data (“Supervised and Unsupervised Machine Learning Algorithms,”2017. The aim of doing so is to be able to determine the output (Y) from the given input (X).

$$Y = f(X)$$

Supervised learning problems use regression and classification techniques for its prediction models

- **Classification**: A classification problem is when the output variable is a category. It is used in predicting disconnected responses, for instance, emails being genuine or not, diseases present or not, etc. and classify them into different categories. Its applications include speech recognition, medical imaging, etc.

- **Regression**: A regression problem is when the output variable is a real value. It predicts based on continuous responses from the input. For example, changes in fluctuations or temperature in power demand. Typical applications include electricity load forecasting and algorithmic trading.

Some examples of supervised machine learning algorithms are:

- Support vector machines
- Random forest
- Linear regression.

2.1.2 Unsupervised Machine learning

This an ML technique used in finding hidden patterns within a given dataset. No labels or classifications are placed on the input data. The system does not always know the right output, but it gets responses from the hidden patterns observed in the unlabeled data ("What is Machine Learning? A definition - Expert System," 2013.).

The main aim of this technique is for the system to be able to discover trends in the input data. There is little or no supervision when using this technique. The algorithms determine the supposed output of the system on their own.

A good example of this type of technique is clustering and association in data analysis. Clustering can be used in grouping datasets or purchase similarities amongst individuals. Association problems arise when rules that are to be used for a huge amount of dataset is to be developed. Applications of clustering include gene sequence analysis, market research, and object recognition ("The 10 Algorithms Machine Learning Engineers Need to Know | Simplilearn," 2019)

Some popular examples of unsupervised learning algorithms are:

- K-means for clustering problems.
- Apriori algorithm for association rule learning problems.

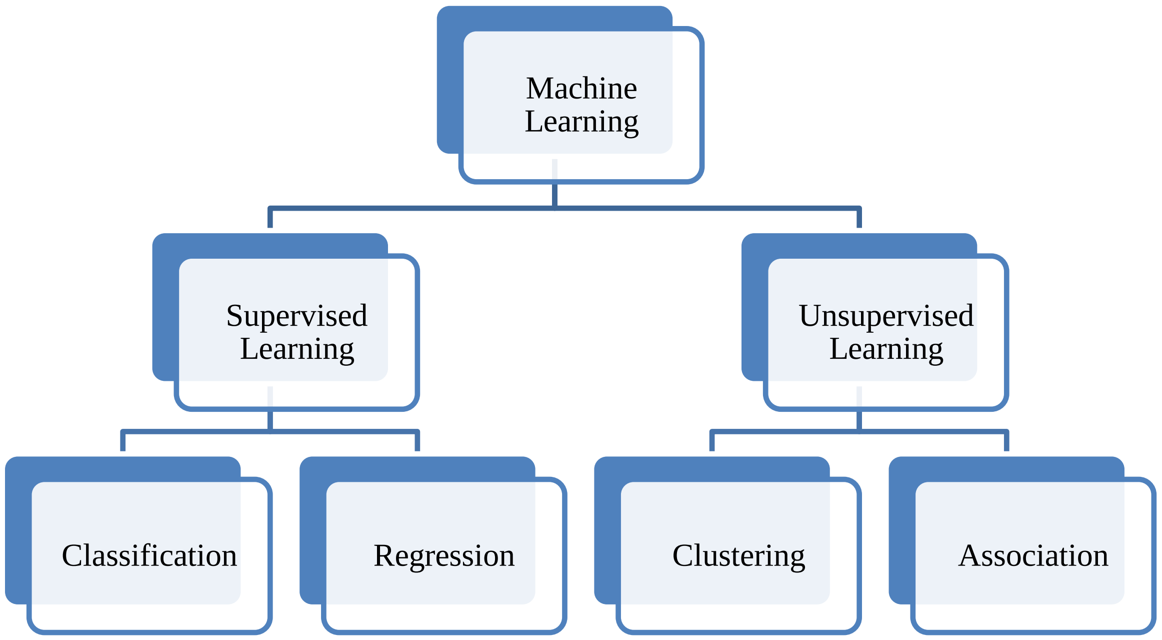


Figure 1: ML hierarchy structure

2.1.3 Applications of Machine Learning

One can consider using ML in solving difficult tasks especially when a huge amount of data and variables are involved without having a solution formula. Below are examples of ML applications in the modern world

i. It is used in facial and speech recognition systems.

ii. The nature of some rising datasets varies, e.g., the financial stock market, shopping trends, etc.; thus, the system needs to adjust to such variations. ML models can be used to classify these variations (MathWorks, 2016).

iii. At the Art and Artificial Intelligence Laboratory of Rutgers University, the researchers tried to classify artworks with respect to genre, artist, and style using computer algorithms. Even though a 100% success was not recorded, the algorithms they developed classified the style of paintings in the database with 60% accuracy, beating the non-expert humans.

iv. Businesses are now able to save time that is often required to manually analyze and predict data. Emails, legal documents etc. can now be trained using machine learning algorithms to generate an output that helps businesses in decision making.

v. Real time analysis to be done as soon as possible as required by an organization can be achieved with the help of machine learning algorithms because of its high precision in real-time.

vi. Because of its ability to avoid human errors such as fatigue boredom etc. ML has been applied in many aspects such as sentiment analysis – topic modeling—language detection etc.

2.2 Literature Review

An automatic trainable summarization procedure based on the application of machine learning has been proposed. It uses a vectorial model and after preprocessing, a sentence is transformed into an N-dimensional vector. Two (2) well-known ML classification techniques were used in this project. They are Naïve Bayes and C4.5 classifiers. C4.5 is an example of a Decision Tree algorithm that's mostly used in comparison with other classification algorithms.

A framework using the ML approach was explored to produce the training text summarizers. This method makes it possible to measure the results of a text summarization algorithm in an objective manner. This avoids the problem of subjective evaluation of the quality of a summary, a topical issue in text summarization research community. Naïve Bayes presented a good result and has since been used in data mining (Neto, Freitas, & Kaestner, 2009).

Initial steps in the text mining process are data collection and preprocessing. They play vital role in the application and techniques of text mining. (C. Cuizon, Lopez, & Rose Jones, 2018) presented the basic steps of preprocessing which are stop words removal, TF/IDF algorithms, and stemming. The authors described stop words as a division of natural language which implies that the removal of words helps to reduce the weight of the text and improve analysis. Removing the stop words helps to reduce the dimensionality of the space in the text. Examples of stop words are, a, an, the, with, etc. They are not regarded as keywords in text mining. The authors also described different stemming methods that can be used in removing suffixes from a word and classified them into statistical, truncation, and mixed methods. The N-gram stemmer is an example of statistical based stemmer.

Lexicon based approach uses a dictionary containing words used to determine their sentiment. Lexicon based approach suffers the disadvantage of having sentiment classification that is dependent on the size of the Lexicon. The increase in size led to more approaches and machine learning. The framework used was Apache Spark framework. For migration of the work-intensive and time-consuming process for manual data annotation, it built its set of training data. The training data consists of tweets labeled by lexical-based pattern analyzer. The framework made employed Naïve Bayes classification and decision tree algorithms. It achieved a 100%

precision and accuracy in the decision tree result (Jain, Anuja Prakash; Dandannavar, 2014).

Ikonomakis, Kotsiantis, &Tampakas (2005) compared recent projects on text classification and came to the conclusion that the best and most frequently used classification technique is the SVM, Naïve Bayes, and Neural networks from different techniques and classifiers and all were based on four parameters, which are: recall, f-measure, precision, and accuracy.

Adel M Hamdan (Mohammad, Alwada', & Al-Momani, 2016) conducted a study on text classification. Rather than using the English language, they tested classifiers using Arabic language document datasets. They employed three different techniques which are Neural Networks, Naive Bayes, and SVM. After implementation, they concluded that SVM produced optimal results compared to the other classifiers. With respect to the neural network, they suggest that different feature selection and feature extraction methods can be employed to achieve better results.

CHAPTER 3

3.1 Classification Steps

Text classification problem can be solved using either supervised learning or unsupervised machine learning technique. The main objective of this project is to be able to make intelligent text classification for a given dataset with the help of the available machine learning algorithm. The dataset is labeled and fed to the machine learning algorithm to perform classification. The algorithm was trained with the dataset and to give the desired output to the defined categories.

In this project, we have decided to label our categories into ‘Sports’, ‘Medicals’, ‘Army’, and ‘Religion’ while in the testing stage, the algorithm is given data that is unobserved and classifies them based on the categories it has been trained with.

3.2 Project Requirements

To be able to achieve an end result, the following requirements are needed

- PC with at least 8gb of Ram
- Python idle
- Numerous dataset

Text classification problem can be solved with the help of supervised learning. The objective is: given a training set of pre-classified text, how can a classifier model be built to predict the class of a given test text. While the data is fed to the algorithm for testing, the algorithm has already been trained on the labeled dataset and to give the desired output (the pre-defined categories). During the testing phase, the algorithm is fed with unobserved data and it classifies them into categories based on the training.

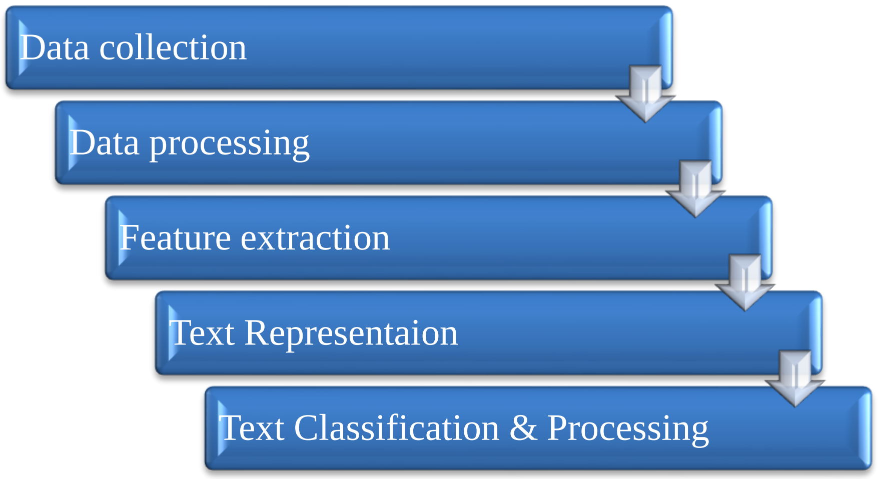
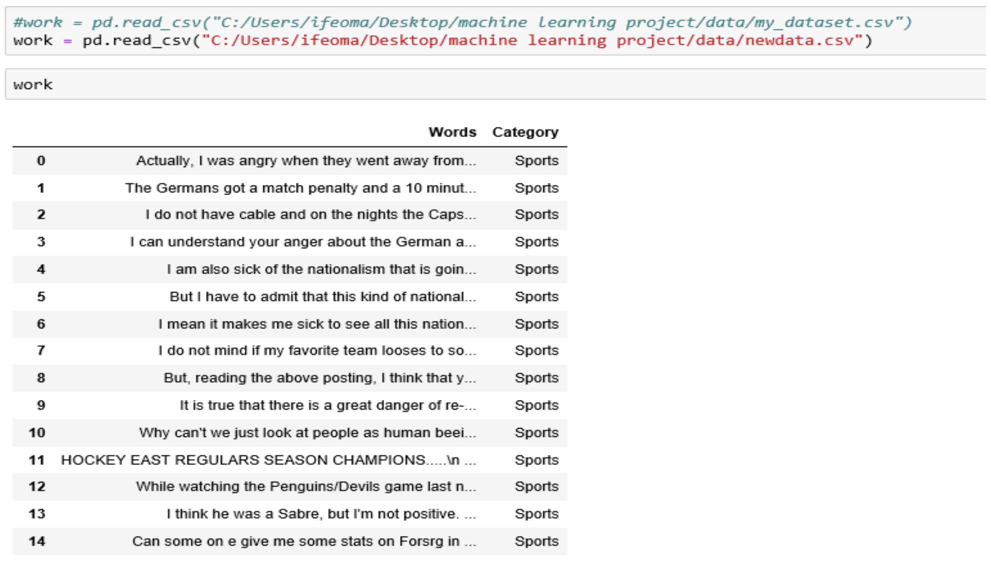


Figure 2: Steps to text classification

3.3 Data collection and preparation

Because of constraint associated with this project, the dataset used for this project was not extremely large. The data was put together from a couple of corpus of different categories obtained from the Internet. This is the basic step in any data mining project a dataset to be used for demonstration must be available. In the prior preprocessing, the dataset took this form



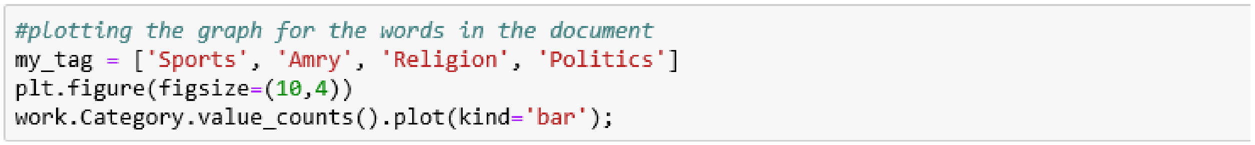
```
#work = pd.read_csv("C:/Users/ifeoma/Desktop/machine learning project/data/my_dataset.csv")
work = pd.read_csv("C:/Users/ifeoma/Desktop/machine learning project/data/newdata.csv")
```

```
work
```

	Words	Category
0	Actually, I was angry when they went away from...	Sports
1	The Germans got a match penalty and a 10 minut...	Sports
2	I do not have cable and on the nights the Caps...	Sports
3	I can understand your anger about the German a...	Sports
4	I am also sick of the nationalism that is goin...	Sports
5	But I have to admit that this kind of national...	Sports
6	I mean it makes me sick to see all this nation...	Sports
7	I do not mind if my favorite team looses to so...	Sports
8	But, reading the above posting, I think that y...	Sports
9	It is true that there is a great danger of re-...	Sports
10	Why can't we just look at people as human beei...	Sports
11	HOCKEY EAST REGULARS SEASON CHAMPIONS.....\n ...	Sports
12	While watching the Penguins/Devils game last n...	Sports
13	I think he was a Sabre, but I'm not positive. ...	Sports
14	Can some on e give me some stats on Forsrg in ...	Sports

Figure 3: Data set description

Before classification, the data to be used has to be preprocessed to remove unwanted characters and symbols. The dataset used contained the following



```
#plotting the graph for the words in the document
my_tag = ['Sports', 'Amry', 'Religion', 'Politics']
plt.figure(figsize=(10,4))
work.Category.value_counts().plot(kind='bar');
```

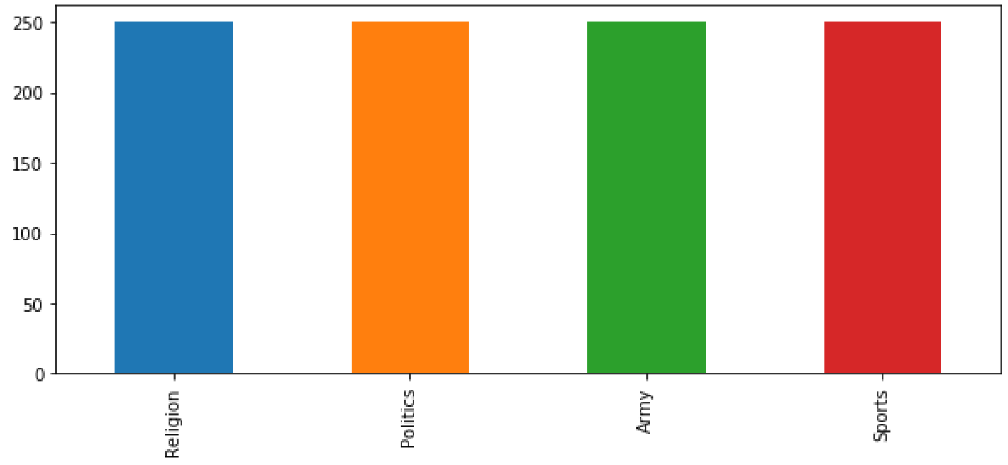


Figure 4: Category visualization

3.4 Preprocessing Data

The dataset used in this project are text data containing noise as seen above. In any machine learning classification, preprocessing of the dataset is an essential step to remove the unwanted characters. There are various steps to preprocess a dataset which are

- **Tokenization**

  This process converts the words or text in each document into tokens of different meanings. The end results of the tokenized words are called tokens. Sentences are being split into tokens.

- **Stop-words Elimination**

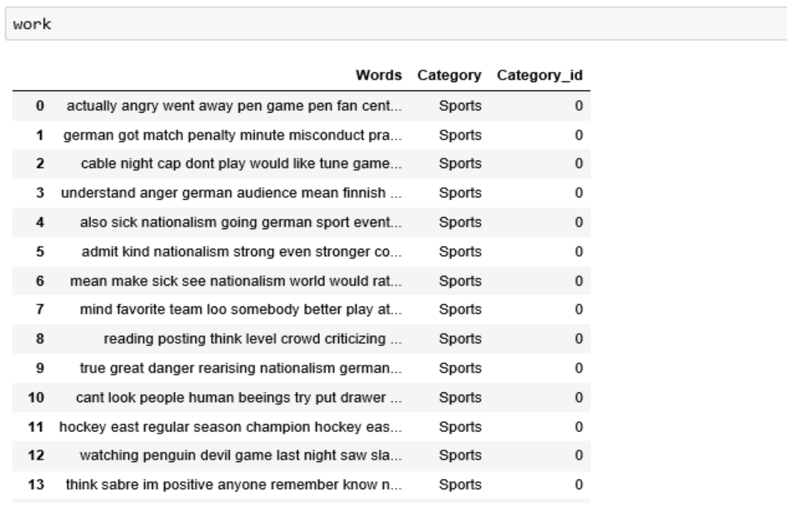
  These are words that are cleaned out of the dataset for the processing of natural text. They are words that do not usually carry meanings in classifications. They are very reoccurring in any language. Common stop words include the following: are, is, a, an, and, then, we, they, there, of, for, no, yes, but, and so forth. The parameter used to discover stop words ensures that all stop words in a text document are removed.

- **Stemming Words Elimination**

  Stemming is the conversion of different words to their root word. For example: Processing – Process, Recession – Recess etc. Removing suffixes automatically is an important operation which is essentially useful in the information retrieval (IF) field.

- **Normalization**

  This is the process of converting the words with upper casing to lower casing.



```
work
```

	Words	Category	Category_id
0	actually angry went away pen game pen fan cent...	Sports	0
1	german got match penalty minute misconduct pra...	Sports	0
2	cable night cap dont play would like tune game...	Sports	0
3	understand anger german audience mean finnish ...	Sports	0
4	also sick nationalism going german sport event...	Sports	0
5	admit kind nationalism strong even stronger co...	Sports	0
6	mean make sick see nationalism world would rat...	Sports	0
7	mind favorite team loo somebody better play at...	Sports	0
8	reading posting think level crowd criticizing ...	Sports	0
9	true great danger rearising nationalism german...	Sports	0
10	cant look people human beeings try put drawer ...	Sports	0
11	hockey east regular season champion hockey eas...	Sports	0
12	watching penguin devil game last night saw sla...	Sports	0
13	think sabre im positive anyone remember know n...	Sports	0

Figure 5: Words after preprocessing

3.5 Feature Extraction

3.5.1 Vectorization

Machine learning algorithms do not work with the raw texts, therefore, the conversion from text to numbers is often done. Sklearn has many libraries that performs vectorization functions that will process the texts within the same function. Vectorization involves the conversion from text to numbers. There are a few important parameters:

- Min_df
- Max_df
- Ngram_range
- Stop_words

Some vectorization libraries used in this project are:

i. **CountVectorizer**

This provides a simple way for tokenization of words by building a list of known words from the document and also encoding the documents using a list of known or common words. We need to fit the training set. This function helps to learn from a list of common or conversant words in the text document. To encode the document, we call the transform() function on which each was vectorized from. When a vector is encoded, the returned value is the length of the already registered list of word. It appears in form of an array of numbers. The numbers in the array represent the number of times a particular word appears in the document.

ii. **TfidfTransformation**

In vector space model, the documents are in vector form to ensure that the machine learning algorithms can be performed on them. An important aspect to note is the term weighting of the vectors. This is a procedure or concept that is associated with every term in the document for value assess. Term weighting is a procedure that takes place during the text indexing process in order to be able to assess the value of each of the document term. Term weighting is the assignment of numerical values to terms that present their importance in a document in order to improve retrieval effectiveness (Reading & Points, 2009).

Term frequency (tf) is an example of term weight which is dependent on the way the words are dispersed. Another example is the inverse document frequency (idf).

It is a weight which relies on the distribution of the words in the database of the document.

This is the transformation of the count matrix to a more normalized tf or tf-idf form. Tf-idf is a combination of tf and idf. It weights word to know how well the word describes a corpus. It gives a positive value to the number of times a particular word appears in the document within a corpus. It gives a negative value for the number of documents that contain a particular word.

Consider term t and document d 2 D, where t appears in n of N documents in D (Fatima, 2017). The TF-IDF function is of the form:

$$\text{TF-IDF}(t, d, n, N) = \text{TF}(t, d) \times \text{IDF}(n, N)$$

3.5.2 Classification Technique

There are so many classification techniques used in text analysis

i. Logistic Regression

Regression analysis is one of the methods used in identifying important variables that can have an impact on a particular topic of interest. This method makes it possible to determine the important factors that should not be ignored because the factors have similar effects on each other. There are two (2) main terms used in regression

- Dependent Variable: This is what one is trying to understand.
- Independent Variable: These are the factors that affect the dependent variables.

To effectively conduct regression analysis, the dependent variable that can be influenced by the independent variable needs to be defined so as to establish the dataset to worked with (“What is Regression Analysis and Why Should I Use It? | SurveyGizmo Blog,” 2019 n.d.).

An aspect of logistic regression is Multinomial Logistic Regression. Because of the multiple categorization (also known as Softmax Remultinominal), we will employ the use of multinomial LR using either a one versus all or one versus one. It is an extension of binary LR (Starkweather & Moske, 2011)

Logistic Regression, like Naïve Bayes, is a probabilities classifier. A function σ(z), is used to create the probability. The graphical representation of Logistic Regression is as shown below

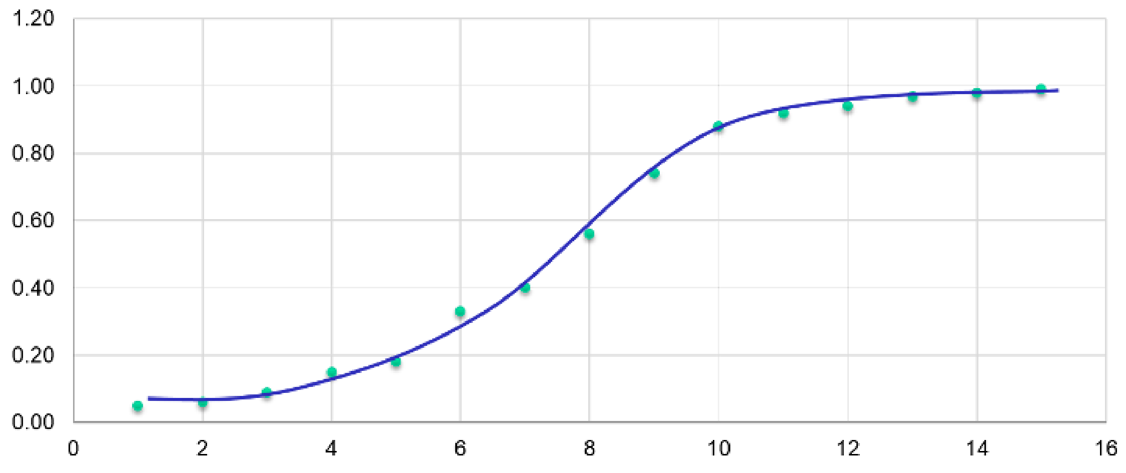


Figure 6: LR Graphical Representation

The function, σ(z) = y, takes a real value between the range 0 and 1. To calculate the logistic regression of a document,

$$y = \frac{1}{1 + e^{-z}}$$

$$z = w.x + b$$

z expresses the weighted sum of the evidence for the class

x represents the vector of features

b represents the interface

To make it a probability, we just need to make sure that the two cases, p(y = 1) and p(y = 0), sum to 1. To achieve this,

$$P(y = 1) = \frac{1}{1 + e^{-(w.x+b)}}$$

$$P(y = 0) = \frac{e^{-(w.x+b)}}{1 + e^{-(w.x+b)}}$$

ii. Naïve Bayes Classifier

This ML classifier is effective, simple to use, and also a popularly used classifier. It makes use of probabilistic classifier decision rule in a Bayesian setting. It is a popular classifier for text classification and for solving spam problems. The main goal of the classifier is to be able to classify features existing in the classes so as to return the most likely class it can belong to.

Bayes rule can be used to achieve this and can be expressed as

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

Its main aim in this project is to classify text into the categories A(A1,A2 … Aj) in line with the text vector gotten from the vectorizer B(B1,B2… Bn). B(B1,B2… Bn) will represent the feature vector of the text while A(A1,A2 … Aj) indicates the categories for the classification. Work can be said to be effectively done for the probability (P1, P2 … Pn) when B(B1,B2… Bn) can be classified to belong to a category A(A1,A2 … Aj) i.e. Pj, being the probability when B(B1,B2… Bn) belongs to Aj.

Naïve Bayes algorithm is possible with the following formula:

P(Aj| B1, B2,..., Bn) =P(B1, B2,... Bn | Aj) * P(Aj)

P(Aj) is the prior probability when the text belongs to Aj

P(B1,B2 … Bn|Aj) is the latter probability that Aj contains the text vector (B1,B2…Bn) when the text to be classified belongs to Aj.

iii. K-Nearest Neighbor (KNN)

K-nearest neighbor deals with the calculated value of k which represents the number of nearest neighbor(s) to be considered. It is referred to as a lazy algorithm where the computations are delayed until classification is done (Bhatia, 2010). It is one of the simplest techniques. The simplest form of KNN is when k = 1. k places a major role in the classification because local region radius is determined by the distance of the kth neighbor. KNN can be referred to as a nonparametric algorithm because it can adjust easily to a nonlinear boundary and eludes assumptions about the class boundary (Bzdok, Krzywinski, & Altman, 2018)

The Euclidean space point is considered in this algorithm. It is the distance between two points within the Euclidean space. The distance between the two points in the plane having p (x,y) and q (a,b) coordinates can be calculated as

$$d(p,q) = \sqrt{(x-a)^2 - (y-b)^2}$$

For classification, the document to be classified will be selected and its categories would be indicated. KNN classification process writes the data in a weight matrix. The k value is determined at the next step so as to get the number of documents to be used as the nearest neighbor to the selected document. The vector distance between the documents is calculated by

$$d(x,y) = \sqrt{\sum_{r=1}^{N} \left(a_{rx} - a_{ry}\right)^2}$$

$d(x,y)$= distance between two documents

N = number of unique words contained in the document

a_{rx} = weight of term r in the document x

a_{ry} = weight of term r in the document y

(Trstenjak, Mikac, & Donko, 2014)

iv. Support Vector Machine (SVM)

It is another type of prominent ML algorithm. It is highly preferred by many people and produces substantial accuracy with fewer computations. It can be used for both classification and regression tasks but mainly used for classification.

Vladimir recommended SVM to be a good binary classifier (Mohammad, Alwada', & Al-Momani, 2016). It is an example of a supervised learning model and can be used in regression analysis and text categorization. In the year 1998, Thorsten Joachims introduced SVM as a means of text categorization and classification (Joachims, n.d.).

The objective of the SVM algorithm is to be able to locate a hyperplane in N-dimensional space (N—the number of features) that has clearly been able to classify the data. In a text document, the texts are represented as vectors where the number of keywords are referred to as dimensions. If the document size is large, then the number of hyper-space dimensions will have an impact in the computation by increasing the cost of the process.

In SVM, the goal is to maximize the distance between the nearest data point and the hyperplane. This is known as margin. Misclassified points are called outliers.

SVM is of two types, namely:

- Linear SVM classifierand
- Non-linear SVM classifier

For the linearly separable data, we make use of the hyperplanes that can separate two or more different classes of data. The distance between the classes is always maximum. The maximum margin hyperplane lies in the middle of the classes.

For the data which can be separated linearly, we select two parallel hyperplanes that separate the two classes of data, so that distance between both lines is maximum. The distance between hyperplanes is referred to as margin. For classification to be close to accurate, maximum margin must exist between the hyperplanes in the middle of the vectors.

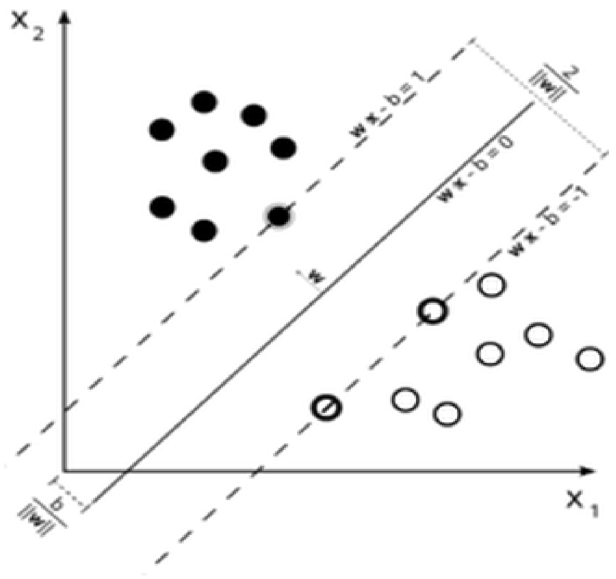


Figure 7: SVM Graphical Representation

$$\vec{w}x_i - b \geq 1 if \Theta i = 1$$

$$\vec{w}x_i - b \leq 1 if \Theta i = -1$$

$|\vec{w}|$ V- represents the normal vector to the plane

Θi – represents the classes

x_i – represents the features

$\frac{2}{\vee\vec{w}\vee}$- represents the distance between two planes

For accurate classification, the equation takes the form

$$|\vec{w}|A\vee \quad {}_{min} for \Theta i(\vec{w}x_i - b) \geq 1\forall_i = 1,2,\ldots n$$

("Svm classifier, Introduction to support vector machine algorithm," n.d.)

CHAPTER 4

4.1 Text Representation

From the previous chapter, we have been able to collect data, preprocess the data, and also extract the information which is in the form suitable for the algorithms. The algorithms used for this project have been discussed previously and their results as well as comparison between the performances of the algorithm with other algorithms will be presented in this chapter.

There are four different categories of classification/categorization. The dataset was split into a training dataset and a test dataset.

Categories	Training Set	Test Set
Sports	188	62
Army	189	61
Religion	183	67
Politics	190	60

Table 1: Data set description

Due to system limitations and frequent power outage, the big data platform was not used for this project as expected.

Count vectorization and tf-idf transformation was implemented as shown in the appendix. The dataset was divided in such a way that 25% was randomly selected and used as the test data (x_test) and 75% was randomly selected and used as the training dataset (x_train). 12974 represents the number of features in the data set.

4.2 Test Classification and Preprocessing

Various libraries such as sklearn, NLTK, pandas, numpy etc. were used to accomplish the end result of the project. The comparison result was based on precision, f1-score, recall and support of the confusion matrix. Confusion matrix is a matrix that gives the performance of the model as the output. The confusion matrix has four (4) major parameters namely

- **True Positive (TP)** – These are the actual predicted true values of the classification. It simply means that the predicted class is 1 (yes) and the actual class is 1 (yes).
- **True Negatives (TN)** – These are the true predicted values that are negative. It simply means that the predicted class is 0 (no) and the actual class is 0 (no).
- **False Positives (FP)** – These are the values that are recorded as false instead of true. It simply means that the predicted class is 1 (yes) and the actual class is 0 (no).
- **False Negatives (FN)** – These are the values that are recorded as true instead of false. It simply means that the predicted class is 0 (no) and the actual class is 1 (yes).

In form of a matrix, the result of the confusion matrix takes this form

	Predicted Class		
		1 (yes)	0 (no)

Actual Class	1 (yes)	True Positive (TP)	False Negative (FN)
	0 (no)	False Positive (FP)	True Negative (TN)

Table 2: Confusion Matrix

4.2.1 Precision

Precision has to do with the positive values in the matrix. It is the ratio of the correctly predicted value to the sum of the predicted positive values. It gives the result of the actual prediction returned by the classifier

$$\text{Precision} = TP / (TP + FP)$$

4.2.2 Recall

Recall is also referred to as sensitivity. It is the ratio of the appropriately predicated positive value to the sum of the actual class 1 (yes). It gives the result of the actual labels to the true positive value. It also means the total percentage of the true positive values correctly classified

$$\text{Recall} = TP / (TP + FN)$$

4.2.3 F1 score

F1 score takes into consideration both false positive values and false negative values. It is the weighted average of recall and precision by two (2). F1 reaches its best value at 1 and its worse at 0. It tells how precise the classifier is.

$$\text{F1 score} = 2 * (\text{Recall} * \text{Precision}) / (\text{Recall} + \text{Precision})$$

4.2.4 Accuracy

Accuracy is one of the most vital results from the classifier. It is the ratio of the total predicted true values to the sum of the total interpretations on the matrix table.

There is maximum output when there is equal number of samples belonging to the various categories

Accuracy = TP+TN / (TP+FP+FN+TN)

4.3 Results

Based on the above, the results from the different classifiers will be stated individually and compared

4.3.1 Multinomial Naïve Bayes (MNB)

This is a very good text classifier compared with its other Bayesian classifiers e.g. Gaussian Naïve Bayes classifier. The Multinomial Naïve Bayes is very fast and accurate in generating the results of some natural language processing applications. Below is the outcome of this text classification algorithm.

Category	Precision (%)	F1 score (%)	Recall (%)	Support (%)
Sports	0.86	0.82	0.79	62
Army	0.79	0.77	0.75	61
Religion	0.91	0.90	0.90	67
Politics	0.83	0.88	0.95	60
Average / Total	**0.85**	**0.85**	**0.85**	**250**

Table 3: MNB Precision, F1 score & Recall

Accuracy = 0.848%

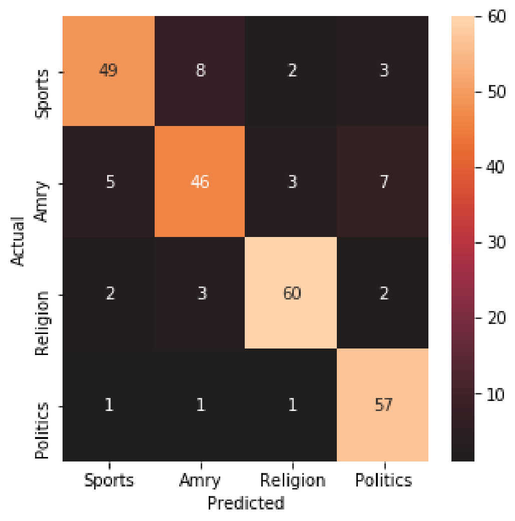


Figure 8: Confusion Matrix for MNB

4.3.2 Logistic Regression (LR)

Logistic Regression has proven to be good for text categorization. Compared to NB algorithm, logistic regression estimates result in a discriminative manner. It has a high accuracy when the dataset to be trained is plenty. However, Naive Bayes can have an advantage when the training data size is small (Davis & Offord, 1997).

Categories	Precision (%)	F1 score (%)	Recall (%)	Support (%)
Sports	0.79	0.80	0.81	62
Army	0.75	0.76	0.77	61
Religion	0.95	0.91	0.87	67
Politics	0.92	0.94	0.97	60
Average / Total	**0.85**	**0.85**	**0.85**	**250**

Table 4: LR Precision, F1 score & Recall

Accuracy = 0.852%

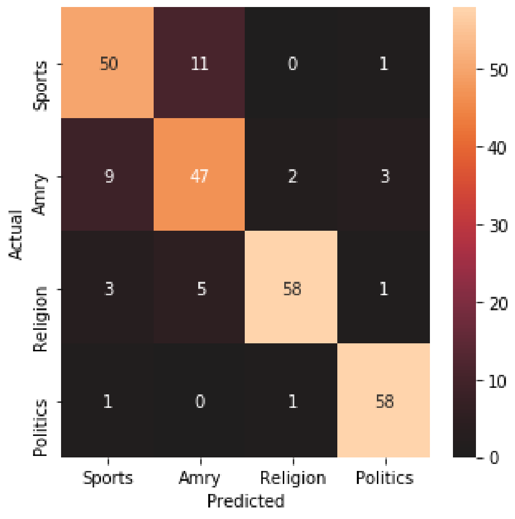


Figure 9: Confusion matrix for LR

4.3.3 Support Vector Machine (SVM)

SVM is a powerful algorithm. It has many kernels that can be used for different applications. The most suitable for this project is the linear kernel because of its linear parameters. Below is the result obtained

Category	Precision (%)	F1 score (%)	Recall (%)	Support (%)
Sports	0.77	0.82	0.82	62
Army	0.77	0.78	0.79	61
Religion	0.94	0.92	0.90	67
Politics	0.98	0.97	0.95	60
Average / Total	0.87	0.87	0.86	250

Table 5: SVM Precision, F1 score & Recall

Accuracy = 0.864%

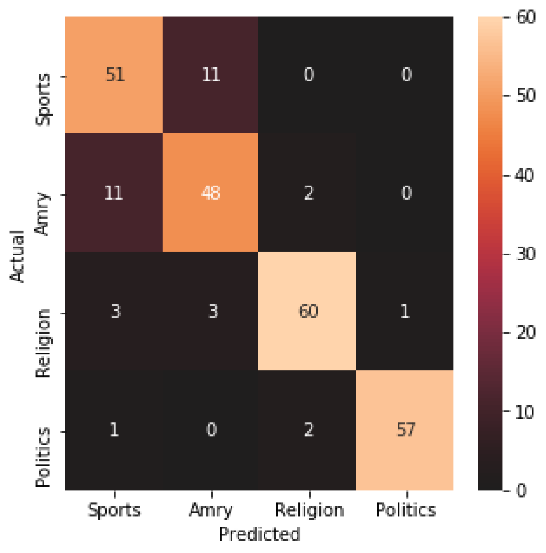


Figure 10: Confusion matrix for SVM

4.3.4 K-Nearest Neighbor (KNN)

KNN is also a text classification algorithm. It is very effective for various problems including text categorization (Guo, Wang, Bell, & Bi, 2006). When tested with datasets, below were the results

Category	Precision (%)	F1 score (%)	Recall (%)	Support (%)
Sports	0.73	0.75	0.77	62
Army	0.84	0.69	0.59	61
Religion	0.88	0.90	0.91	67
Politics	0.76	0.83	0.92	60
Average / Total	**0.80**	**0.80**	**0.80**	**250**

Table 6: KNN Precision, F1 score & Recall

Accuracy = 0.80%

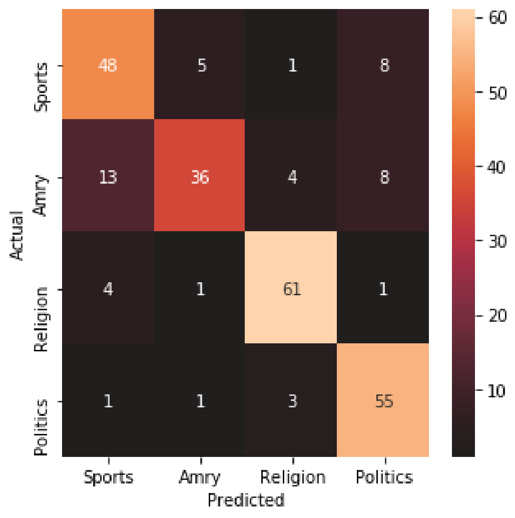


Figure 11: Confusion matrix for KNN

4.4 Comparison Result

Classifier	Accuracy (%)	Precision (%)	F1 Score (%)	Recall (%)
MNB	0.848	0.85	0.85	0.85
LR	0.852	0.85	0.85	0.85
SVM	0.864	0.87	0.86	0.87
KNN	0.8	0.8	0.8	0.8

Table 7: Comparison Table

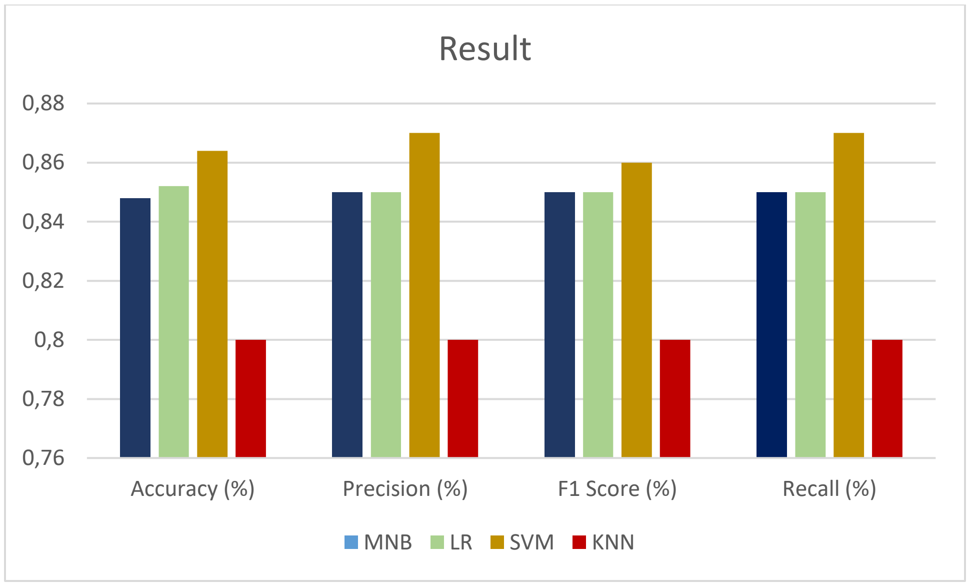


Figure 12: Algorithm Results

CHAPTER 5

5.1 Conclusion

In this project, we have been able to classify text into their various categories as specified. The four (4) Machine Learning algorithms used, MNB, SVM, LR, and KNN were applied on the data set used for this project. This makes it possible to be able to place documents in their rightful position. A comparison between the results obtained using the algorithm was performed to determine which out of the four algorithms performed better. With the help of some python libraries available for analysis, it is clear that SVM outperformed all other algorithms. A major factor to be considered is feature extraction and selection as they play vital roles in obtaining the result.

5.2 Challenges

Gathering of data set was a huge challenge. Although, not many data set was use, but a big data set could have provided a high accuracy.

5.3 Future Works

Our future work would focus on discovery of other methods for extraction and selection whilst implementing the algorithms in a big data platform.

REFERENCES

An easy introduction to Natural Language Processing. (n.d.).

Bhatia, N. (2010). Survey of Nearest Neighbor Techniques. In *IJCSIS) International Journal of Computer Science and Information Security* (Vol. 8).

Bzdok, D., Krzywinski, M., & Altman, N. (2018). Machine learning: supervised methods. *Nature Methods*, *15*(1), 5–6. https://doi.org/10.1038/nmeth.4551

Davis, L. J., & Offord, K. P. (1997). Logistic regression. *Journal of Personality Assessment*, *68*(3), 497–507. https://doi.org/10.1207/s15327752jpa6803_3

Dr. S. Vijayarani, Ms. J. Ilamathi, M. N. (2018). *Preprocessing Techniques for Text Mining*. *5*(1), 7–16. https://doi.org/10.1016/j.procs.2013.05.286

Fatima, S. (2017). Text Document categorization using support vector machine. *International Research Journal of Engineering and Technology (IRJET)*, 141–147.

Guo, G., Wang, H., Bell, D. A., & Bi, Y. (2006). *Using kNN model for automatic text categorization Using k NN Model-based Approach for Automatic Text Categorization*. (March). https://doi.org/10.1007/s00500-005-0503-y

Joachims, T. (n.d.). *Text Categorization with Support Vector Machines: Learning with Many Relevant Features*.

Mohammad, A. H., Alwada', T., & Al-Momani, O. (2016). Arabic Text Categorization Using Support vector machine, Naïve Bayes and Neural Network. *GSTF Journal on Computing (JOC)*, *5*(1), 108–115. https://doi.org/10.5176/2251-3043_4.4.360

Nene, S. (2017). *Deep Learning for Natural Language Processing*. 930–933.

Reading, R., & Points, K. (2009). *Table Design ▶*.

Saranya, K., & Jayanthy, S. (2018). Onto-based sentiment classification using machine learning techniques. *Proceedings of 2017 International Conference on Innovations in Information, Embedded and Communication Systems, ICIIECS 2017*, *2018-January*, 1–5. https://doi.org/10.1109/ICIIECS.2017.8276047

Starkweather, J., & Moske, A. K. (2011). *统计(讲原理)-Multinomial Logistic Regression (lecture notes)*. *51*(6), 404–410. https://doi.org/10.1097/00006199-200211000-00009

Structured vs. Unstructured Data. (n.d.).

Svm classifier, Introduction to support vector machine algorithm. (n.d.).

Text Analytics: A Primer | GreenBook. (n.d.).

Trstenjak, B., Mikac, S., & Donko, D. (2014). KNN with TF-IDF based framework for text categorization. *Procedia Engineering*, *69*, 1356–1364. https://doi.org/10.1016/j.proeng.2014.03.129

What is Regression Analysis and Why Should I Use It? | SurveyGizmo Blog. (n.d.).

Appendix

```
#!/usr/bin/env python
# coding: utf-8

import pandas as pd
import numpy as np
import re
import matplotlib.pyplot as plt
from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize



#work = pd.read_csv("C:/Users/ifeoma/Desktop/machine learning project/data/my_dataset.csv")
work = pd.read_csv("C:/Users/ifeoma/Desktop/machine learning project/data/newdata.csv")

#creating columns for the category_id
col = ['Words', 'Category']
work = work[col]
work = work[pd.notnull(work['Words'])]
print(work['Words'].apply(lambda x: len(x.split(' '))).sum())
def Cat(i):
    if (i == 'Sports'):
        return 0
elif (i == 'Amry'):
        return 1
elif (i == 'Religion'):
        return 2
    else:
        return 3
```

```
work['Category_id'] = work['Category'].apply(Cat)


# In[5]:


#plotting the graph for the words in the document
my_tag = ['Sports', 'Amry', 'Religion', 'Politics']
plt.figure(figsize=(10,4))
work.Category.value_counts().plot(kind='bar');


#before preprocessing
def print_plot(index):
    example = work[work.index == index][['Words', 'Category']].values[0]
    if len(example) > 0:
        print(example[0])
        print('Tag:', example[1])
print_plot(521)

#preprocessing
from bs4 import BeautifulSoup
import string
import nltk
#nltk.download('wordnet')
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
from nltk.stem import PorterStemmer
space_replace = re.compile('[/(){}\[\]\|@,;]')
```

```
regular_symbols = re.compile('[^0-9a-z # +_]')
stopWords = set(stopwords.words('english'))



def clean_text(text):
    """
        text: a string


        return: modified initial string
    """
    text = BeautifulSoup(text, "lxml").text # HTML decoding
    text = text.lower() # lowercase text
    text = re.sub(r'\d+', '', text) #removes numbers
    text = text.strip() #removes white spaces
   # text = text.translate(string.maketrans(" ',' "), string.punctuation)
    text = space_replace.sub(' ', text) # replace REPLACE_BY_SPACE_RE symbols by space in
text
    text = regular_symbols.sub('', text) # delete symbols which are in BAD_SYMBOLS_RE from
text
    text = ' '.join(word for word in text.split() if word not in stopWords) # delete stopwors from text


    text = word_tokenize(text) #tokenize
    #text = " ".join(text)
    #text = [i for i in text if not i in stopWords]



lem=WordNetLemmatizer()
    #a = []
    text = " ".join(lem.lemmatize(word) for word in text)
    #for i in text:
        #print (len(i))
```

```
       # if (i == ' '):
        #text = a.append(i)
        #else:
        #text = lem.lemmatize(i)

        #text = a.append(t)
    #text = word_tokenize(text)
    #st = PorterStemmer()
    #for word in text:
     #   text = st.stem(word)
    return text

work['Words'] = work['Words'].apply(clean_text)
print_plot(521)

stemmer = PorterStemmer()
stops = set(stopwords.words("english"))

for i in range (len(work)):
   for row in work:
stemmed_first = ""
       c = 0

       for i in work:
           if c <len(work)-1:
stemmed_first += stemmer.stem(i) + " "
           else:
stemmed_first += stemmer.stem(i)
           c += 1
           #stemmed_second = ""
        #c = 0
        #for w in list2:
         #   if c <len(list2)-1:
          #      stemmed_second += stemmer.stem(w) + " "
           # else:
            #    stemmed_second += stemmer.stem(w)
            #c += 1

        #print stemmed_first
        #print stemmed_second
plt.scatter(work['Words'], work['Category'])
Category = ['Sports', 'Amry', 'Religion', 'Politics']

x = work['Words']
y = work['Category']
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.25,random_state=42)

cv = CountVectorizer(ngram_range=(1, 4))
x_train = cv.fit_transform(x_train)
x_test = cv.transform(x_test)
tf_transformer = TfidfTransformer(use_idf=True).fit(x_train)
x_train = tf_transformer.transform(x_train)
#x_train = x_train.toarray()

tf_transformer = TfidfTransformer(use_idf=True).fit(x_test)
x_test = tf_transformer.transform(x_test)
#x_test = x_test.toarray()
```

```
print(x_train.shape, x_test.shape)


# In[13]:


type(x_train)


# In[30]:


from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.25,random_state=42)
features = ['Trump', 'God', 'Christian', 'gun', 'fight', 'army', 'foul', 'yellow card']
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(encoding = 'utf-8', decode_error = 'ignore', analyzer = 'word', vocabulary = features)
x_train = cv.fit_transform(x_train)

analyze = cv.build_analyzer()
feature_names = cv.get_feature_names()
trans_array = x_train.toarray()
type(trans_array)
#print(trans_array)


# In[15]:
```

```
from sklearn.metrics import r2_score #accuracy
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import TfidfTransformer

from sklearn import metrics



# In[16]:



mnb_clf = MultinomialNB(alpha=0.4)



mnb_clf.fit(x_train, y_train)
y_pred = mnb_clf.predict(x_test)
print('accuracy %s' %metrics.accuracy_score(y_pred, y_test))
print(classification_report(y_test, y_pred,target_names=Category))
print(confusion_matrix(y_test, y_pred))
#visualize_classifier(mnb_clf , x_test, y_test);



# In[20]:



import seaborn as sns
```

```
conf_mat = confusion_matrix(y_test, y_pred)

ax = plt.subplots(figsize=(5,5))

sns.heatmap(conf_mat, annot=True, center = 0, fmt='d', xticklabels=Category,
yticklabels=Category)

plt.ylabel('Actual')

plt.xlabel('Predicted')

plt.show()



# In[21]:



#testing using Regression

from sklearn.linear_model import LogisticRegression

from sklearn.pipeline import Pipeline

from sklearn.metrics import r2_score #accuracy

from sklearn.metrics import classification_report


logreg = LogisticRegression(n_jobs=1, C=1e5)

logreg.fit(x_train, y_train)

lr_pred = logreg.predict(x_test)

print('accuracy %s' %metrics.accuracy_score(lr_pred, y_test))

print(classification_report(y_test, lr_pred,target_names=Category))

print(confusion_matrix(y_test, lr_pred))



# In[22]:

import seaborn as sns

conf_mat = confusion_matrix(y_test, lr_pred)

ax = plt.subplots(figsize=(5,5))
```

```
sns.heatmap(conf_mat, annot=True, center = 0, fmt='d', xticklabels=Category,
yticklabels=Category)

plt.ylabel('Actual')

plt.xlabel('Predicted')

plt.show()



# In[23]:

from sklearn import svm

SVM = svm.SVC(C=1000.0, kernel='linear', degree=8, gamma='auto')

SVM.fit(x_train,y_train)

prediction_SVM = SVM.predict(x_test)

#svm_clf = svm.SVC(kernel = 'rbf', C=1000.0, max_iter=3000)

#svm_clf.fit(x_train,y_train)

#svm_y_pred = svm_clf.predict(x_test)

print('accuracy %s' %metrics.accuracy_score(prediction_SVM, y_test))

print(classification_report(y_test, prediction_SVM,target_names=Category))

print(confusion_matrix(y_test, prediction_SVM))



# In[24]:

import seaborn as sns

conf_mat = confusion_matrix(y_test, prediction_SVM)

ax = plt.subplots(figsize=(5,5))

sns.heatmap(conf_mat, annot=True, center = 0, fmt='d', xticklabels=Category,
yticklabels=Category)

plt.ylabel('Actual')

plt.xlabel('Predicted')

plt.show()

from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier()
```

```
knn.fit(x_train, y_train)

knn_pred = knn.predict(x_test)

print('accuracy %s' %metrics.accuracy_score(knn_pred, y_test))

print(classification_report(y_test, knn_pred,target_names=Category))

print(confusion_matrix(y_test, knn_pred))

import seaborn as sns

conf_mat = confusion_matrix(y_test, knn_pred)

ax = plt.subplots(figsize=(5,5))

sns.heatmap(conf_mat, annot=True, center = 0, fmt='d', xticklabels=Category,
yticklabels=Category)

plt.ylabel('Actual')

plt.xlabel('Predicted')

plt.show()
```