# Implementation, testing and validation report for Social Media Post Sentiment Classifier System

DOCUMENT NO: 1.0

PREPARED BY: BSE 19-5

DATE: 31-05-2019

VERSION: 1.0

*Document Approval*

| ***Name*** | ***Role*** | ***Date*** | ***Signature*** |
|---|---|---|---|
| *BSE 19-5* | *Author(s)* | | |
| *NTANDA MOSES* | *Validation* | | |
| | *Client* | | |

# Table of Contents

## CHAPTER 1: INTRODUCTION

Overview: The following chapter aims to describe the background, significance, scope of the Social Media Post Sentiment Classifier and explain the techniques used throughout the system development. A broad definition will be given for the core concepts involved in the development of the artifacts: Natural language processing, Machine learning and Text Mining. Furthermore, specialized terminologies and the graphical user interface will be explained.

### 1.1 Background.

As internet is growing bigger, its horizons are becoming wider. Social Media and Micro blogging platforms like Facebook and Twitter dominate in spreading news and trending topics across the globe at a rapid pace. A topic becomes trending if more and more users are contributing their opinion and judgments, thereby making it a valuable source of online perception. These topics generally intended to spread awareness or to promote public figures, political campaigns during elections, product endorsements and entertainment like movies, award shows. Large organizations and firms take advantage of people's feedback to improve their products and services which further help in enhancing marketing strategies. Thus, there is a huge potential of discovering and analyzing interesting patterns from the infinite social media data for business-driven applications.

The intention is to gain an overview of the wider public opinion behind certain topics. Precisely, it is a paradigm of categorizing conversations into positive, negative or neutral labels. Sentiment analysis and opinion mining, due to its social and commercial value, has become a very hot topic of research these days

Hence sentiment analysis and user opinion mining on online social media has a great social and commercial importance. In this research, a framework is proposed to analyze Facebook posts and comments for opinions and sentiments of the public.

In the last three years, sentimental analysis has become a hot trend topic of scientific and market research in the field of Natural Language Processing and machine learning[2].

Extracting the public opinion from social media text provides a challenging and rich context to explorer computational models of natural language processing.

### 1.2 Problem Statement.

Sentiment analysis of reactions to social media posts is still a challenge. This is as a result of language grammatical error, some comments are vague due to the use of slangs, sentiment and subjectivity are quite context-sensitive, and, at a coarser granularity, quite domain dependent. Due to the above-mentioned challenges, we aim at analyzing textural data and build a sentiment classifier tool that is able to label reactions as positive, negative or neutral.

### 1.3 Significance.

The product will be used as backbone for some application developers. Our project will enable organizations and companies analyze massive feedback from people's reactions on social media.

Political parties may be interested to know if people support their program or not and this project will help solve this problem.[3] This would reduce the costs spent on employing people to analyze these reactions manually.

### 1.4 Scope.

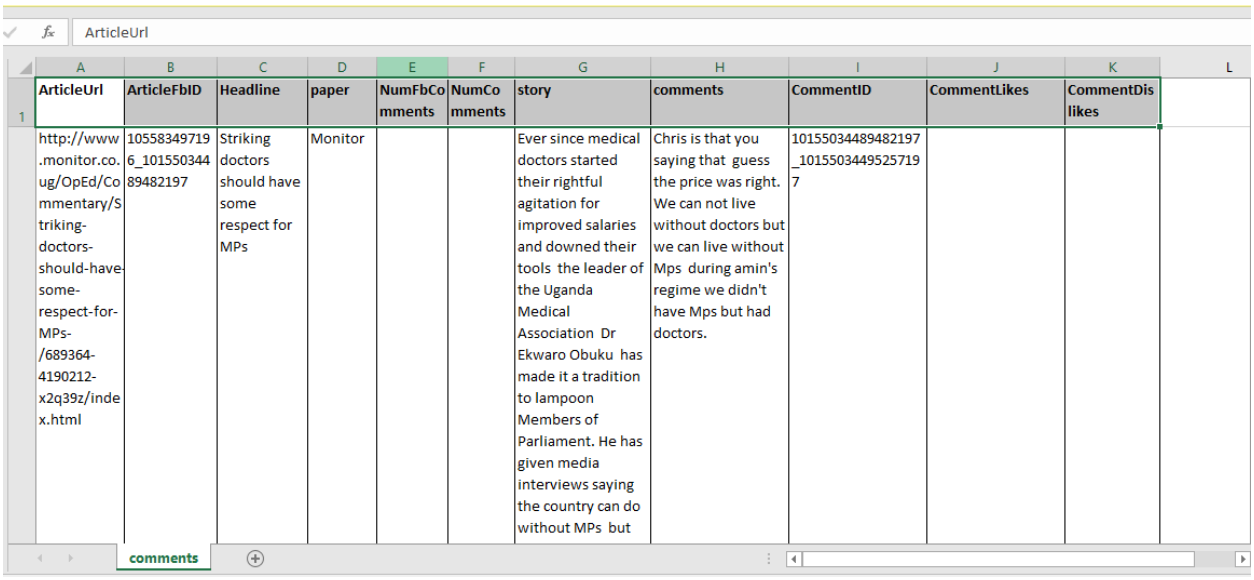
Our project aims at analysis of reactions to social media posts on Facebook. The study is going to be carried out on reactions to posted articles by Daily Monitor, New Vision, Observer and Red Pepper on Facebook. The project will cover comments only in English. Comments in any other language will not be covered in this project. The research development and implementation of this project is in the duration that is stipulated for the final year project development.

### 1.5 Classification model background

#### 1.5.1 Loading the Data

As usual, we first downloaded our datasets locally, and then we loaded them into data frames using Python. A total of 4435 posts and approximately 120160 comments were downloaded from

the Facebook API, this resulted from a condition of only extracting posts with at least 5 comments within a time range 02/06/1995 to 19/02/2019.



| ArticleUrl | ArticleFbID | Headline | paper | NumFbComments | NumComments | story | comments | CommentID | CommentLikes | CommentDislikes |
|---|---|---|---|---|---|---|---|---|---|---|
| http://www.monitor.co.ug/OpEd/Commentary/Striking-doctors-should-have-some-respect-for-MPs-/689364-4190212-x2q39z/index.html | 105583497196_10155034489482197 | Striking doctors should have some respect for MPs | Monitor | | | Ever since medical doctors started their rightful agitation for improved salaries and downed their tools the leader of the Uganda Medical Association Dr Ekwaro Obuku has made it a tradition to lampoon Members of Parliament. He has given media interviews saying the country can do without MPs but | Chris is that you saying that guess the price was right. We can not live without doctors but we can live without Mps during amin's regime we didn't have Mps but had doctors. | 10155034489482197_10155034495257197 | | |

Figure 1 .1 Data extraction from Facebook

### 1.5.2 Preparing the Corpus

In linguistics, a corpus or text corpus is a large and structured set of texts (nowadays usually electronically stored and processed). They are used to do statistical analysis and hypothesis testing, checking occurrences or validating linguistic rules within a specific language territory. In our particular case, we are talking about the collection of reaction fragments that we want to classify either positive, negative or neutral. Working with text corpora involves using natural language processing techniques. Python is capable of performing really powerful transformation with textual data. However, we used just some basic ones.

The requirements of a bag-of words classifier are minimal in that sense. We just need to count words, so the process is reduced to do some simplification and unification of terms and then count them. The simplification process mostly includes removing punctuation, lowercasing, removing stop-words, and reducing words to its lexical roots (i.e. stemming).

So, we process our reactions and create a corpus. We also extract important words and establish them as input variables for our classifier.

### 1.5.3 Text Mining.

Text mining refers to the analysis of data contained in natural language text. (E.g. posts and reactions retrieved from Facebook). It can be defined as the practice of extracting meaningful knowledge from unstructured text sources. The application domain of text mining varies from research, sentiment analysis and marketing applications among others.

In research, text miming is used to find out opinions of people about particular issues, in marketing, text mining is relevant to the analysis of the customer relationship management where the company can improve their predictive analysis models for customer turnover by keeping track of the customer's opinions.

The main goal of text mining is to process data into a structured format ready for analysis, via application of natural language processing and other analytical methods [1]. There are many aspects with in the field of study of text mining and information extraction relevant for this project. Consequently, the following material aims to explain the challenges and terminology associated with information extraction and subsequent processing.

### 1.5.4 Natural Language Processing.

The data retrieved from Facebook presents amount of unstructured data and they do not follow any formal structure and neither will they be grammatically correct. It is also expected that abbreviations and short forms of words, as well as slang will be encountered in the text analyzed. Moreover, sentences describing the same or similar ideas may have very different syntax and employ different vocabularies [1]. Given a post and a reaction, a predefined textural format has to be produced at processing time. The techniques presented below were used in the development of the project.

#### 1.5.4.1 Tokenization.

The first task, that must be completed before any processing can occur, is to divide the textual data into smaller components. This is a common step in a Natural Language Processing (NLP) application, known as tokenization. At a higher level, the text is initially divided into paragraphs and sentences. In these regards, the project aim at this step is to correctly identify sentences. This can be done by interpreting the punctuation marks such as a period mark ".", within the text analyzed. The next step is to extract the words (tokens) from sentences. The challenge at this step

is to handle the orthography within a sentence. Consequently, spelling errors have to be corrected; URLs and punctuation shall be excluded from the resulting set of tokens. After tokenizing a reaction, the returned result is an array containing a set of strings.

#### 1.5.4.2 Part of Speech Tagging.

In order to understand the complete meaning of a sentence, the relationship between its words has to be established. This can be done by assigning every word a category that identifies syntactic functionality of that word. Also known as part of speech tagging (POS), this step can be seen as an auxiliary requirement for n-grams selection and lemmatization. Table 1 covers the part of speech notations used in the project.

Table 1 Part of speech tags used throughout the project

| | |
|---|---|
| ADJ: adjective | PART: particle |
| ADV: adverb | PRON: pronoun |
| AUX: adjective | PROPN: proper noun |
| CONJ: conjunction | PUNCT: punctuation |
| DET: determiner | SYM: symbol |
| NOUN: noun | VERB: verb |
| NUM: numeral | X: other |

#### 1.5.4.3 Stemming and Lemmatization.

The goal of both stemming and lemmatization is to reduce inflectional forms and derivations of a word to a common base form [2]. For example, the following words: “connection”, “connections”, “connective”, “connected”, “connecting” will have the same base, which is “connect”. Stemming, is a crude heuristic process that chops off the ends of words, so that only the base form is kept [2]. By contrast, lemmatization uses the morphological analysis of the words, returning their dictionary form (base), commonly referred as the lemma. However, for a language like English as opposed to more morphologically rich languages, this process relies on a dictionary being available. In addition, a lemmatizer can introduce ambiguity by proposing all possible lemmas for a word form or by choosing the wrong proposal from two competing lemmas (e.g., is axes the plural of axe or of axis?). Considering the arguments presented above,

algorithm chosen for this task is Porter's stemming algorithm. It consists of five phases, where word reductions are performed. For each phase, rules and conventions to apply them are defined [3].

Table 2 Stemming rules and examples - Porter [2]

| Rules | Examples |
|---|---|
| SSES -> SS | caresses -> caress |
| IES -> I | ponies -> poni |
| SS -> SS | caress -> caress |
| S -> / | cats -> cat |

#### 1.5.4.4 N-grams.

N-grams is a common technique in text mining, where word subsets of length n within a sentence are formed. From the sentence "This is a six words sentence!" the following n-grams can be formed:

1-grams (unigrams): "this", "is", "a", "six", "words", "sentence"

2-grams (bigrams): "this is", "is a", "a six", "six words", "words sentence"

3-grams (trigrams): "this is a", "is a six", "a six words", "six words sentence"

As such, the example sentence above will produce 6 unigrams, 5 bigrams, and 4 trigrams. On a bigger data set, producing bigrams and trigrams will considerably contribute to the size of the data set, consequently, slowing down the system.

#### 1.5.4.5 Converting Text to Numbers

Machines unlike humans, cannot understand raw text. Machines can only see numbers. Particularly, statistical techniques such as machine learning can only deal with numbers.

Therefore, we need to convert our text into numbers.

Different approaches exist to convert text into corresponding numerical form. The Bag of Words Model and the Word Embedding Model are two of the most commonly used approaches. In this project, we use the bag of words model to convert our text to numbers.

Bag of Words

We use both the Counter vectorizer. It contains the max_features parameter which is set to 1200. This is because when you convert words to numbers using the bag of words approach, all unique words in all documents are converted to features. All the documents can contain tens of thousands of unique words. But the words that have a very low frequency of occurrence are usually not a good parameter for classifying documents.

Therefore, we set the max_features parameter to 1200, which means that we want to use 1200 most occurring words as features for training our classifier. The next parameter is min_df and it has been set to 5. This corresponds to the minimum number of documents that should contain this feature. So, we include only those words that occur in at least 5 documents. Similarly, the max_df, feature is set to 0.7; in which the fraction depends to a percentage. Here 0.7 means that we should include only those words that occur in a maximum of 70% of all documents. Words that occur in almost every document are not usually not suitable for classification because they do not provide any unique information about the document.

### 1.5.5 Machine Learning Classification.

The rest of this section will present machine learning algorithms used to classify the polarity (positive, negative, neutral) of the reactions in their normalized form. The term machine learning refers to the "automated detection of meaningful patterns in data" [3]. Alongside with the growing size of the data produced, machine learning has become a common technique for information extraction. From spam filtering and personalized advertising, to search engines and face detection software, machine learning is applied in a wide range of domains [3]. While the variety of present algorithms depends on the learning task, specialized literature makes this distinction according to the nature of interaction between the computer and the environment. As such, the separation is made between supervised and unsupervised machine learning algorithms:

Supervised Learning: In a supervised machine learning algorithm the training data "comprises examples of the input vectors along with their corresponding target vectors(classes)" [4]. For example, in a supervised learning manner a computer can be thought to distinguish between pictures of cats and pictures of dogs. In the training phase, a set of labeled pictures will be processed by the algorithm. At this point, the computer 'knows' which pictures contain cats and which contain dogs. When presented with new unlabeled pictures, the algorithm will decide based on what it 'saw' before, the type of animal in the picture. Hence, the goal is to 'learn' a general rule that maps input to output [4].

Unsupervised Learning: Unsupervised machine learning algorithms have the same scope as supervised learning, which is to map input to output. However, the difference is that in the training phase the input is not labeled, consequently, the computer has to find structure in the input, without specifically being told how to classify. As part of the project, a supervised approach was desirable. Consequently, two algorithms were used, one implemented and the other taken from a pre-implemented library which serves as a comparison base for the evaluation process. The rest of this chapter will explain in detail the Logistic Regression (implementation is explained in Chapter 4), and offer a brief description of the Naïve Bayes algorithm.

### 1.5.6 Naïve Bayes

Naïve Bayes is a supervised machine learning algorithm. It is widely recognized as one of the "most efficient and effective learning algorithms for data mining" [5]. The classifier is built using Bayes theorem with independence assumptions. As such, the classifier assumes that the effect of a predictor (x) over a given output class (y) is independent of the value of other predictors [4].

$$P(C/X) = ((P(X/C).\ P(C)) / P(X)$$

Such that; P(C/X) = Posterior probability of the target (X) given a predictor (C).

P(X/C) = probability of a predictor given a class (C).

P(C) = prior probability of a class

P(X) = prior probability of a predictor.

To illustrate the formula, consider the example with images of cats and dogs where a computer can be thought to distinguish between pictures of cats and pictures of dogs. In the training phase, a set of labelled pictures will be processed by the algorithm. At this point, the computer 'knows' which pictures contain cats and which contain dogs. In this case, the classifier has to decide between two classes. Classifying a new image is the equivalent of comparing between the two probabilities: P (cats | new_image) and P (dogs | new_image). These values can be computed based on the above formula. In the computation, the images used in the training phase of the algorithm will be used to compute the probability of the two classes, the prior probability of the predictors, and the probability of a predictor given the two classes.

Also known as class conditional independence, the aforementioned assumption is often seen as a drawback in the accuracy of the algorithm. A number of heuristics meant to handle this challenge will be presented in the Implementation chapter. To exemplify the formula, consider the previous example with images of cats and dogs. In this case, the classifier has to decide between two classes. Classifying a new image is the equivalent of comparing between the two probabilities:

P (cats | new_image) and P (dogs | new_image). These values will be computed based on the above formula. In the computation, the images used in the training phase of the algorithm will be used to compute the probability of the two classes, the prior probability of the predictors, and the probability of a predictor given the two classes.

### 1.5.7 Logistic Regression.

Logistic regression predicts the probability of an outcome that can only have two values. The prediction is based on the use of one or several predictors (numerical and categorical). Logistic regression describes data and explains the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables. Below is an example of logistic regression equation.

$$y = e^{\wedge}(b_0 + b_1 * x) / (1 + e^{\wedge}(b_0 + b_1 * x)).$$

Such that; Y is the predicted output, $b_0$ is the bias or intercept term and $b_1$ is the coefficient for the single input value (x). Each column in the input data has an associated b coefficient that must learned from the training data [6].

### 1..5.8 Feature Extraction.

Feature extraction is the process of building a feature vector from a given comment with respect to a post. Each entry in a feature vector is an integer that has a contribution on attributing a sentiment class to a comment. This contribution can vary from strong, where the value of a feature entry heavily influences the true sentiment class; to negligible, where there is no relationship between feature value and sentiment class. It is often the job of classification algorithm to identify the dependency strength between features and classes, making use of strong correlated features and avoiding the use of 'noisy features'.

The output of this process is feature vector. This process extracts the features of both the comments and the post. The training and new data features will later be split. The features will be used to build the model. The model is saved in a data store. The user on the other hand enters new data for classification. The classifier uploads the trained and built models. The uploaded model classifies the new data. The classifier outputs the results. If the output data is not correct, the user updates it. The updated comments are stored into a database. These updated comments are later used to update the model. This helps the model to learn.

### 1.5.9 Training of Classifier:

We built our feature vector using bag of words, social media and lexical feature sets as described above. Once the feature vector is built for each instance in the training set, these vectors are then fed to the classifier (learning algorithm). We used Logistic Regression as our classification algorithm. This method analyzes feature vectors with a labeled class to identify dependency relationship between each feature and a sentiment. To illustrate this process briefly: each vector is considered as data point in vector space of dimension equal to feature vector size.

When a new unlabeled input (post and comment) is provided to the system, it extracts the feature vector in the same manner as it did for labeled data. Finally, the vector is given as an input to the learned model.

Note that this process would account for two class classification whereas our problem of sentiment classification contains three classes. To handle that, Logistic regression model from

the new user classification where a user is given an option to sentiment in their perspective. Using this technique, the model is updated and able to learn on how to sentiment even the neutral sentiments.

### 1.5.10 Algorithm Used

We built the classifier using the Multinomial Naïve Bayes Classifier. Naïve Bayes classifier is based on Bayes theorem which uses the theorem to build a probabilistic classifier. We basically sidelined with Naïve Bayes classifier because it is simple, trains faster than other classifiers, and can work well when we don't have sufficient training data. We can also create solid baselines with little efforts and help us to explore more complex solutions. We also tried Logistic Regression however, Naïve Bayes yielded the best results.

### 1.5.11 Performance Measures Used

After designing the model, we needed to find out how good the model is at labeling users' reactions. This section was very important because it delineated how good our predictions were. We used a confusion matrix to describe the performance of the classification model on the set of data for which the true values were known as shown in the table below.

Table 3 showing how confusion matrix is generated.

| Actual | | | |
|---|---|---|---|
| Predicted | | Positives | Negatives |
| | Positives | TP | FP |
| | Negatives | FN | TN |
| | | | |

True Positives (TP) and True Negatives (TN) are the observations that are correctly predicted. We minimized False Negatives (FN) and False Positives (FP).

True Positives: These are the correctly predicted values which mean the value of actual class is yes and the value of the predicted class is also “yes”. E.g. when a user comments a positive reaction and the predicted class tells you the same thing.

True Negatives: These are the correctly predicted negative values which means that the value of actual class is also “no”. E.g. when a user comments a negative reaction and the predicted class tells you the same thing.

False Positives: These come up when the actual class is no and the predicted class is yes. E.g. the user comments a negative reaction and the model classifies it as positive.

False Negatives: These come up when the actual class is yes but the predicted class is no. E.g. the user comments a positive reaction and the model classifies it as negative. Basing on these four parameters, we measured the performance of the classifier using Accuracy and Precision methods.

### 1.5.12 Accuracy.

This is the ratio of the correctly predicted score to the total observations. Accuracy can be got from the equation below.

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

We used accuracy because the target variable classes in the data are nearly balanced.

### 1.5.13 Recall.

Recall can be achieved as a percentage of positive case found. Generally, from formula,

Recall = (TP) / (TP+FN).

### 1.5.14 Precision.

This is the ratio of correctly predicted positive observations to the total predicted positive observations. The question this metric answer is of all reactions that were labeled as positive, how many were actually positive?

Precision = (TP) / (TP+FP).

We also used precision because we want to be precise on how good our model works.

### 1.5.15 F1-SCORE

F1-score is a measure of test's accuracy. It is interpreted as the weighted average of precision and recall. It considers both the precision and the recall of the test to compute the score: precision is the number of correct positive results divided by the number of all positive results returned by the classifier, and recall is the number of correct positive results divided by the number of all relevant samples (all samples that should have been identified as positive).

F1 = 2 * (precision * recall) / (precision + recall)

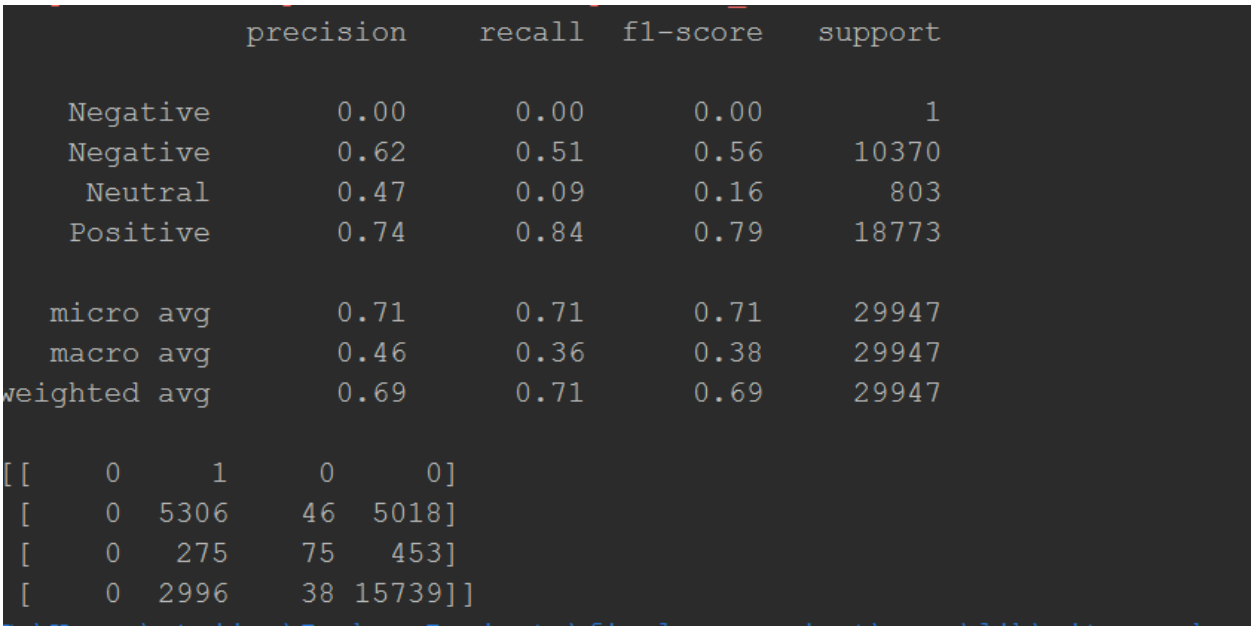
### 1.5.16 Evaluation and Results

In this chapter a detailed evaluation of the engine is provided, as well as accuracy comparison between the main classification algorithms implemented – Naïve Bayes, and the Linear Regression algorithm.

#### 1.5.1.6 Evaluation

Classifiers Comparison. In the initial stages of the project, research in the field of machine learning was imperative to decide which classification algorithm was best suited for implementation [14] [6]. Consequently, in the final version of the engine, the Linear Regression (LG) was implemented, whilst the second choice remained the Naïve Bayes algorithm (NB). Table 3.1 shows an accuracy comparison between the two algorithms, based on the features employed.

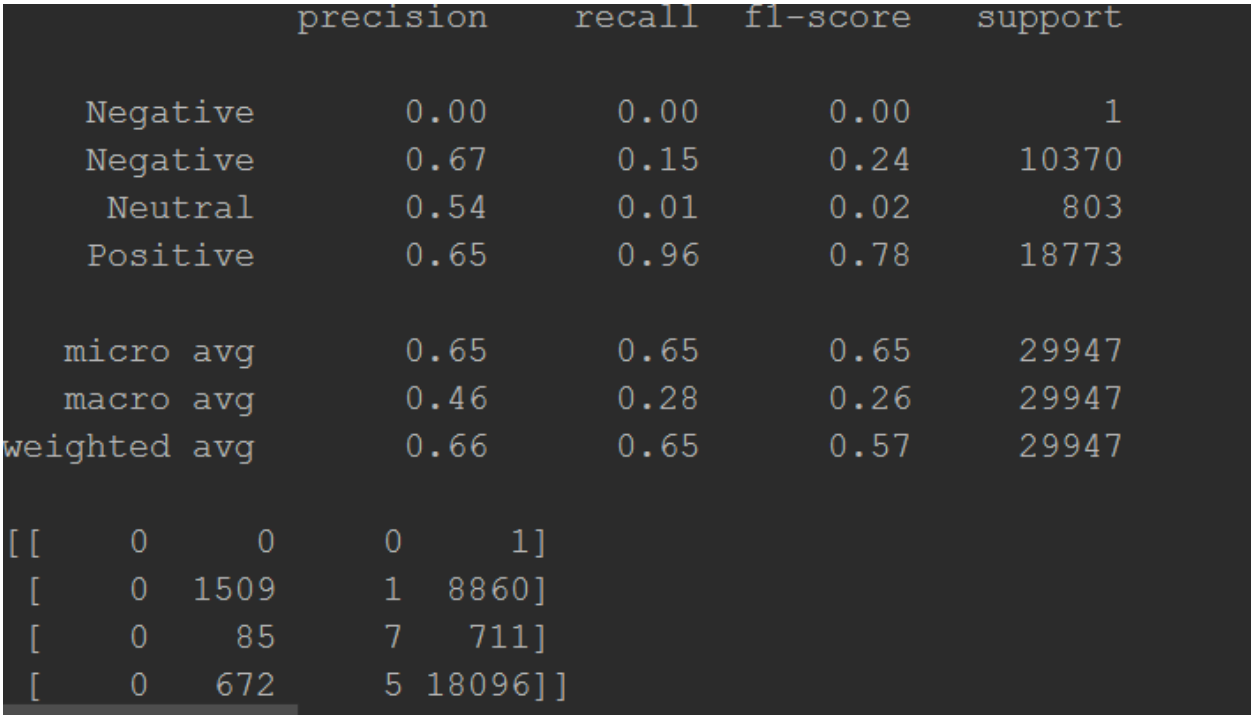
Required: Confusion matrix for both algorithms to be placed here.



| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Negative | 0.00 | 0.00 | 0.00 | 1 |
| Negative | 0.62 | 0.51 | 0.56 | 10370 |
| Neutral | 0.47 | 0.09 | 0.16 | 803 |
| Positive | 0.74 | 0.84 | 0.79 | 18773 |
| micro avg | 0.71 | 0.71 | 0.71 | 29947 |
| macro avg | 0.46 | 0.36 | 0.38 | 29947 |
| weighted avg | 0.69 | 0.71 | 0.69 | 29947 |

```
[[    0     1     0     0]
 [    0  5306    46  5018]
 [    0   275    75   453]
 [    0  2996    38 15739]]
```

Figure 1. 1 Logistic Regression confusion matrix



| | precision | recall | f1-score | support |
|---|---|---|---|---|
| Negative | 0.00 | 0.00 | 0.00 | 1 |
| Negative | 0.67 | 0.15 | 0.24 | 10370 |
| Neutral | 0.54 | 0.01 | 0.02 | 803 |
| Positive | 0.65 | 0.96 | 0.78 | 18773 |
| micro avg | 0.65 | 0.65 | 0.65 | 29947 |
| macro avg | 0.46 | 0.28 | 0.26 | 29947 |
| weighted avg | 0.66 | 0.65 | 0.57 | 29947 |

```
[[    0     0     0     1]
 [    0  1509     1  8860]
 [    0    85     7   711]
 [    0   672     5 18096]]
```

Figure 1. 2 Naïve Bayes confusion Matrix

The classifier was able to achieve a classification accuracy of 80% which is above our target accuracy (75%)

While all the other components (Interface and Application) have been built using tools specific to the components as described:

Generally, the following are the steps required to create a sentiment classification model in Python:

Importing Libraries

Importing The dataset

Text Preprocessing

Converting Text to Numbers

Training and Test Sets

Training Text Classification Model and Predicting Sentiment

Evaluating the Model

Saving and Loading the Model

Upload the model for end user to perform classification.

**1.6 Overview of the document.**

This document describes the implementation, testing and validation findings for the Social Media Post Sentiment Classifier system. It is divided into the following sections:

Chapter 1: This section gives an overview of the document, the scope and background of the project.

Chapter 2: The section describes and specifies the system completely and is the basis for the validation process. It describes the inputs, outputs of the system and describes the functionalities of the proposed project together with difficulties faced during implementation.

Chapter 3: This section describes the design and implementation of the system.

Chapter 4: This chapter presents the testing methodologies that were applied during development.

Chapter 5: This chapter describes the installation process, input and supplementary files used during the system installation.

Chapter 6: This section describes the performance measures that were considered throughout the project development process.

Chapter 7: This is a reflective chapter, where the completion of the objectives set at the start of the project is assessed. In addition, future work is proposed and the knowledge gained while working on the project is demonstrated.

## Chapter 2: System Specifications.

The requirements describe and specify the system completely and are basis for the development and validation process.

### 2.1 version of requirements and version control.

The requirements of the Social Media Posts Sentiment Classifier underwent through a series of verifications, validation checks and approvals for a proper implementation strategy. These changes were recommended due to thorough relevancy and compatibility in various versions of the Software Requirements Documents

We used GitHub to manage the system.

These include the following;

Table 4 Requirements Specifications

| Version | Requirements | Description changes | Author/Proposed | Day |
|---|---|---|---|---|
| Version1.0 | First Requirements Document | Initial document | BSE19-5 | 6-8 |
| Version 1.1 | Problem statement | Change of the problem statement to reflect the problem at hand | BSE19-5/ Mr. Ntanda Moses | 1-2 |
| Version 1.2 | System name | Change of system name to reflect the scope of our study | Mr. Ntanda Moses | 2-3 |
| Version 2.0 | System features | 1. add in Admin panel to handle and manage users<br>2. Allow a user to upload a csv file to classify | Mr.Ntanda Moses | 6-8 |

| | | | | |
|---|---|---|---|---|
| | | many posts at once | | |

**2.2 Input**

The user requires several sets of inputs to render its operation. These inputs include;

- **Username and Password**: This acts as the first and major constrain for users to access the full functionality of the system. Both the admin and the client user require to log in by usernames and passwords. The username and password are recorded during the registration process prior to access of the system features.

  In case of the incorrect username and password, message is displayed to the user to try again or retrieval change password in case of forgotten password.
- **User details:** The input is majorly to provide information about the user accessing the system.
- **Csv with post and comment, direct post and comment:** A user can either input a csv file containing standardized posts and comments or a direct post and comment. Once they have all been input, the data passes through preprocessing and feature extraction process which derives other inputs for the preprocessing functions. These derived inputs include, tokens, part of speech tags, stems and tokens and features.

**2.3 Output.**

The system displays several outputs to users depending on the request and the processing made by system component and features

- Registration and subscription forms: This is the form on which a user or an admin can register and login for the effective use of the system.
- Subscription details: The admin can view information recorded during the subscription of the users to access the system as retrieved from database.

- Visualizations: This is any technique for creating images, diagrams animations to communicate a message. To produce output, SPSC uses visualization to summarize the large contents of inputs.

  Line and bar graphs were used for visualization in order to present the summary of sentiments produced categorized amongst labels/classes.

Other derived outputs include;

Tokens, stemms, part of the speech tags, features, lamments a confusion matrix. These outputs are also used as inputs in preprocessing processes.

### 2.4 Functionality.

Most of the requirements for the engine are of a functional nature. Hence, heuristics on improving the machine learning algorithm were required, as well as the use of third-party software. An initial list of functional requirements ranked by priority and days required for completion is presented in Table 3.1. In order to achieve the behavior desired, a Facebook API had to be used to collect data that was used to develop the classifier.

#### 2.4.1 Functional Requirements.

Table 5 Functional Requirements

| No | Requirements | Priority | Days | Development Stage |
|---|---|---|---|---|
| 1 | Train the main classification algorithm | Very High | 5-7 | Early |
| 2 | Output and store the model after the training phase of the classification algorithm | Very High | 1-2 | Early |
| 3 | Test the classification algorithm proposed | Very High | 1-2 | Early |
| 4 | Clean the noise from the data | High | 2-3 | Middle |

| | | | | |
|---|---|---|---|---|
| | mined | | | |
| 5 | Store the acquired data depending on the component in which the engine is used | High | 2-3 | Middle to late |
| 6 | Train an additional machine learning algorithm for comparison with the main algorithm | Medium | 6-8 | Late |
| 7 | Based on the selected topic and a reaction, classify the reaction as negative, positive or neutral | Medium | 4-5 | Late |

Another important requirement was data storage which was implemented with Sqlite3.

### 2.4.2 Graphical User Interface

As one of the main objectives of the project is performing highly accurate sentiment analysis, the graphical user interface was not within the initial scope. However, a graphical user interface turned out to be imperative, such that users can interact with the algorithm/classifier in a more intuitive manner. As such, a set of functional requirements was developed in the GUI as evidenced in Table 6

Table 6 GUI Requirements

| No | Functional Requirements | Priority | Development stage |
|---|---|---|---|
| 1 | Upload CSV file of post and reactions and classify the reactions | Very High | Late |
| 2 | Show the label of the intended reaction | High | Late |
| 3 | Include text area (input field) | Very High | Late |

| | | | |
|---|---|---|---|
| | where data to be classified is entered | | |
| 4 | Perform a general summary of the classification | High | Late |
| 5 | Show visualization (graphs) | High | Late |
| 6 | User registration | High | Late |
| 7 | User login | High | Late |
| 8 | Admin view and manager users | High | Late |
| 9 | Admin upload csv to update model | Very High | Late |

The non-functional requirements for the graphical user interface were set following three principles: responsiveness, readability, and usability. As such, Table 1.4 presents the non-functional requirements developed for the graphical user interface.

Table 7 GUI Non-Functional Requirements

| No | Non-functional requirements | Type |
|---|---|---|
| 1 | The graphical user interface should have a quick response time for user interactions | Responsiveness |
| 2 | The graphical user interface should present an intuitive design. | The Readability |
| 3 | The graphical user interface should output the text body of the reactions analyzed | Usability |

**2.5 Limitation and Safety.**

Table 8 Limitations and Safety

| | |
|---|---|
| Limitations | For a user to know the sentiment of a comment with respect to its post, the use has to input both a post and a comment.<br><br>For a user to perform a classification of posts |

| | |
|---|---|
| | and comments in a csv file, the columns in the csv file must be in order of Post, Comment respectively.<br>Limitation on the client user on what he/she can view or not view.<br><br>The user accessing the system must register with a valid email address.<br><br>The user must have an internet connection to access the system and perform a classification.<br><br>The system is built with many packages of which package versions change over time which might affect the system.<br><br>User registration inputs are validated as required so that users cannot skip any input.<br><br>Users of the system must be computer literate as well as equipped with social media knowledge since our target users were social media handlers.<br><br>Only inputs (post) from one social media platform (Facebook).<br><br>The Facebook post must be from four Uganda publishing companies (Daily Monitor, New Vision, Observer and Red Pepper). |

| | |
|---|---|
| | Only English language posts are considered for classification.<br><br>Access is limited to computers. It cannot be accessed on mobile devices. |
| Safety | • The font used by the system is clear and support for maximizing is enabled to support easy reading and prevent eye straining.<br>• The system has been built using a well proven framework Flask to ensure easy testability, maintainability.<br>• Extensive data validation has been done to prevent system failure due to wrong user input.<br>• The system is not physical and holds no physical implication on a human.<br>• The csv file must contain only Post, Comment columns in the Post, Comment order.<br>• The validation inputs are validated by required so that users cannot skip any input. |

**2.6 Default settings**

By default, the system allows itself to be updated when users confirm that the returned classification is actually the actual classification as the user expected.

By default, the system uses the porter stemmer for stemming words. In the next releases, the user will be provided with an option of selecting their own stemmer.

By default, the system uses stop words provided by the NLTK library. The user will be provided with an option of specifying their own stop words in the next releases.

After power up of a computer connected to the internet, we assume there is installed supported web browser that will allow one to access the system once it is hosted online.

The admin login credentials are default. The admin doesn't have to register because the username and password are default. He can later change the login credentials later.

### 2.7 Special Requirements

### 2.8 Errors and Alarms.

The possible errors of the system and their triggers include:

**Invalid credentials:** This is an error caused by entering inputs that are not valid with respect to the required data, for example, when registering into the system. These errors are handled through check of compatibility of the system of which soft notifications requesting the user to re-enter a particular data that is not valid.

**Not CSV file:** When performing a classification, a csv file must be uploaded and an error will be raised if a file other than a CSV file is uploaded.

**Wrong order of CSV fields:** The system requires one to enter a csv file in the order of Post, Comment so that classification can be made. Any input with a wrong order or wrong columns raises an error

## Chapter 3: Design Output.

### 3.1 Implementation (coding and compilation)

The system has been developed using different tools that have been evaluated to suite the requirement needs of the specific components.

Python language was used to develop the Classifier component because it would easily provide generators, which are particularly useful when processing large amounts of data, passing the source data through the processing chain, one item at a time, storing only the results of the processing chain [8]. We used python 3.7.2 for implementing our system.

Why Python?

Given the context of processing large amounts of data, memory management became a proprietary. Consequently, a programming language which is capable to handle processing and storage of these amounts of data was imperative. An iterative system approach, when processing a list of items, has to firstly store it, which requires memory. In these regards, Python provides generators, which are particularly useful when processing large amounts of data, passing the source data through the processing chain, one item at a time, storing only the results of the processing chain [8].

Considering the above argument, Python proves capable of efficiently managing memory, a task crucial for the 'real time' component of the project. A drawback of Python is that it is an interpreted language, which by contrast, is slower than compiled languages (such as C, Java, etc.). The developers' community considered the disadvantage, and proposed different ways to improve Python's speed. As such, projects like Numba and PyPy are viable solutions. The creator of Python, Guido van Rossum recognizes the improvements added to Python and states that PyPy is the best way to obtain high-performance systems while using Python [8].

Furthermore, advanced libraries for data processing such as Scikit-learn, Pandas, Keras, NumPy, SciPy and among others were developed by the scientific community and domain experts [9]. Such tools proved to be helpful during the development stage, and reinforced the reasoning of choosing Python.

Database/Back end.

This included database designing. We designed it using SQLite3.

Interface Component.

This included software development tools suitable for interface development, business logic implementation. We used FLASK, HTML, CSS and JavaScript for developing the user interface.

**FLASK**: Flask is a third-party micro framework in Python, which offers a build-in development server with restful request dispatching [7]. "Micro" in micro framework means Flask aims to keep the core simple but extensible. Flask stands out from other frameworks because it lets developers take the driver's seat and have full creative control of their applications. Flask supports relational databases, which was needed to save our classified data.

**HTML**: We used html to develop static web pages for the user interface.

**CSS**: We used CSS for page formatting and design.

**JavaScript:** JavaScript enables interactive web pages and thus is an essential part of our web application product.

Other development tools used to implement the system include;

- PyCharm community 2018.2.3 used as an IDE for coding.
- Flask framework used to develop web pages
- GitHub was used in version control.
- Web browsers such as chrome and Firefox to display the web pages.
- Facebook API was used to get test and training data for classifier model.
- NLTK library which supports machine learning with algorithms like naïve Bayes, logistic regression. It also contains packages which were used in the development process like TF-IDF vectorizer and Counter vectorizer.

### 3.2 Design changes

There were some design changes made when implementing the system. Below are the changes made on the design of the system.

Table 9 Variations made on the design compared to what was specified in the requirements

| Component | Design change |
|---|---|
| Upload CSV file | The initial requirement was that user inputs only a post and a comment and a classification is made but then there might be a need for a user to classify many social posts with many comments and this component was added to cater for that requirement. |

| | |
|---|---|
| Admin | Initially, the system's main requirement was to perform labelling of posts and comments as either negative, positive or neutral but later we realized there is a need to manage users of the system. |

### 3.2.1 Design change justification

Uploading the CSV with many posts and comments will allow the user to classify as many posts and comments as they want as this would ease the work of trying to analyze posts and comments one by one.

For proper usage purposes, we thought there was a need to come up with an administrator requirement to manage those who can use the system.

## 3.3 Documentation.

The documents that have been provided as output from the Design include;

- User Manual: This guides the user on how to use the system, which operations to perform and their resultant results.
- Design Document: This helps the user to understand fully implementation of the project.

Table 10 Design Details

| Topics | Design output |
|---|---|

| Topics | Design output | |
|---|---|---|
| Good programming practice | Source code is...<br>☑ Modulized<br>☑ Encapsulated<br>☑ Functionally divided<br>☐ Strictly compiled<br>☑ Fail-safe (handling errors) | Source code contains...<br>☑ Revision notes<br>☑ Comments<br>☑ Meaningfull names<br>☑ Readable source code<br>☐ Printable source code |
| Dynamic testing | ☑ All statements have been executed at least once<br>☑ All functions have been executed at least once<br>☑ All case segments have been executed at least once<br>☑ All loops have been executed to their boundaries<br>☐ Some parts were not subject to dynamic test<br>Comments: | |

## Chapter 4: Inspection and Testing.

### 4.1 Introduction

This chapter presents the testing methodologies that were applied during development, under the test-driven development recommendations [14]. As such, unit testing was exhaustively used for each component module of the engine and the graphical user interface. In addition, end to end testing was performed on the major parts of the system. The resultant system was tested with the

users to see if it works as expected and to provide the stake holders with information about the quality of service proposed. It was aimed at meeting the business and technical requirements that guided design and development.

Before implementing the new system into operation, a test run of the system was done removing all the errors that existed. It is an important phase of a successful system. After codifying the whole programs of a system, a test plan was carried out and run on a given set of test data. The output of the test run was matched with the expected results. System testing is considered as part of implementation process.

### 4.2 Test plan and performance

The test plan is created during the development or reverse engineering phase and identify all elements that are about to be tested. The test plan should explicitly describe what to test, what to expect, and how to do the testing. Subsequently it should be confirmed what was done, what was the result, and if the result was approved.

#### 4.2.1 Test types and objectives

This section presents the testing methodologies that were applied during development, under the test-driven development recommendations [14]. As such, through functional testing (feature testing), unit testing was exhaustively used for each component module of the engine and the graphical user interface. In addition, end to end testing was performed on the major parts of the system. The resultant system was tested with the users to see if it works as expected and to provide the stake holders with information about the quality of service being proposed. It was aimed at meeting the business and technical requirements that guided design and development.

Before actually implementing the new system into operation, a test run of the system was done removing all the errors that existed. It is an important phase of a successful system. After codifying the whole programs of a system, a test plan was carried out and run on a given set of test data. The output of the test run was matched with the expected results. System testing is considered as part of implementation process.

Using the test data, the following test run are carried out:

#### 4.2.2 Level of tests.

##### 4.2.2.1 Unit testing

For the algorithm, unit testing was done in a systematic manner, each module being tested during or immediately after it was developed. In these regards, “pytest” - a third party library for Python was used. An expected behavior is defined and tested against the input. Consequently, all the modules developed were tested using the same format [12].

This approach was useful in discovering errors which affected the engine. As such, the stream module and the graphical user interface required a significant amount of time for debugging compared to the other modules. While the GUI seemed challenging due to the limited knowledge in JavaScript, the stream module was affected by the use of the Facebook API which for the purpose of the project required a multi-threaded implementation.