**Effective Garbage Data Filtering Algorithm for SNS Big Data Processing by Machine Learning**

**ABSTRACT**

**ABSTRACT**

Recently, as the use of social network services (SNS) increases in daily modern life, the amount of SNS data generated has become very large. In addition, increasing efforts are being directed to extracting various pieces of information by collecting, processing, and analyzing large amounts of SNS data. While various pieces of information can be extracted from SNS data through big data processing, this is a highly resource intensive task. Therefore, to obtain information from SNS data, a lot of time and material resources are required. In this paper, we propose a data filtering algorithm that filters out garbage data that has no meaning as data among SNS data. The proposed algorithm improves the filtering accuracy by recursive learning based on the initial learning data. Experimental results show that the proposed algorithm has a filtering effect of over 70% on the experimental keywords.

TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| S.NO | CONTENT | PGNO |
| 1 | Introduction   * 1. Software requirements   2. Hardware requirements |  |
| 2 | Feasibility study  2.1 Economic feasibility  2.2 Technical feasibility  2.3 Social feasibility |  |
| 3 | Literature survey |  |
| 4 | System analysis  4.1 Existing system  4.1.1 Disadvantages of existing system  4.2 Proposed system  4.2.1 Advantages of proposed system  4.3 Functional requirements  4.4 Non-Functional requirements |  |
| 5 | System design  5.1 System architecture  5.2 UML diagrams |  |
| 6 | Implementation  6.1 Modules  6.2 Sample code |  |
| 7 | Software environment |  |
| 8 | System testing  8.1 Testing strategies  8.2 Test cases |  |
| 9 | Screens |  |
| 10 | Conclusion |  |
| 11 | References |  |

LIST OF FIGURE

|  |  |  |
| --- | --- | --- |
| FIG.NO | FIG.NAME | PG.NO |
| 5.1.1 | System architecture |  |
| 5.1.2 | Flow diagram |  |
| 5.1.3 | Dataflow diagram |  |
| 5.2.1 | Usecase diagram |  |
| 5.2.2 | Class diagram |  |
| 5.2.3 | Activity diagram |  |
| 5.2.4 | Sequence diagram |  |
| 5.2.5 | Collaboration diagram |  |
| 5.2.6 | Component diagram |  |
| 5.2.7 | Deployment diagram |  |

**INTRODUCTION**

**1.INTRODUCTION**

Recently, the number of users of social network services (SNS) is increasing due to the explosive growth of mobile devices, and the amount of data generated on SNS is increasing correspondingly. SNS is widely used for social relations and friendship, but recently, it has been increasingly used for the secondary purpose of gathering and analyzing large datasets on SNS and obtaining various pieces of information. The data on SNS includes content related to opinions being expressed in various fields such as economy, society, and culture. Therefore, by analyzing the data on SNS, information on various flows and opinions on topics such as society, economy, and politics can be extracted. However, it is very difficult and time-consuming to accurately analyze the data on SNS as it consists of a mix between positive data that is helpful to the actual analysis, advertisement data, and irrelevant data. In recent years, as interest in big-data processing has increased, studies have been conducted on collecting and storing big data in a stable manner and more efficiently processing data using limited computing resources. However, less research and fewer studies are available regarding the utility of big data before they are processed. Therefore, this study investigates how to effectively filter garbage data from big data, and thereby improve the accuracy and speed of the data analysis in real big-data processing as Figure 1. In particular, this study focuses on improving the filtering accuracy by including machine learning in the process of filtering garbage data. Therefore, in this study, we propose an algorithm that can improve the garbage data filtering accuracy of SNS big data by cyclic learning and prove the effectiveness of the algorithm through experiments.

**1.1 SOFTWARE REQUIREMENTS**

The functional requirements or the overall description documents include the product perspective and features, operating system and operating environment, graphics requirements, design constraints and user documentation.

The appropriation of requirements and implementation constraints gives the general overview of the project in regards to what the areas of strength and deficit are and how to tackle them.

* **Python idel 3.7 version (or)**
* **Anaconda 3.7 ( or)**
* **Jupiter (or)**
* **Google colab**

**1.2 HARDWARE REQUIREMENTS**

Minimum hardware requirements are very dependent on the particular software being developed by a given Enthought Python / Canopy / VS Code user. Applications that need to store large arrays/objects in memory will require more RAM, whereas applications that need to perform numerous calculations or tasks more quickly will require a faster processor.

* **Operating system : windows, linux**
* **Processor : minimum intel i3**
* **Ram : minimum 4 gb**
* **Hard disk : minimum 250gb**

**FEASIBILITY STUDY**

**2. FEASIBILITY STUDY**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**2.1 ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### **2.2 TECHNICAL FEASIBILITY**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**2.3 SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

**LITERATURE SURVEY**

**3.LITERATURE SURVEY**

**3.1 A survey of machine learning for big data processing:**

**https://asp-eurasipjournals.springeropen.com/articles/10.1186/s13634-016-0355-x**

**ABSTRACT:** There is no doubt that big data are now rapidly expanding in all science and engineering domains. While the potential of these massive data is undoubtedly significant, fully making sense of them requires new ways of thinking and novel learning techniques to address the various challenges. In this paper, we present a literature survey of the latest advances in researches on machine learning for big data processing. First, we review the machine learning techniques and highlight some promising learning methods in recent studies, such as representation learning, deep learning, distributed and parallel learning, transfer learning, active learning, and kernel-based learning. Next, we focus on the analysis and discussions about the challenges and possible solutions of machine learning for big data. Following that, we investigate the close connections of machine learning with signal processing techniques for big data processing. Finally, we outline several open issues and research trends.

**3.2 Big data classification: Problems and challenges in network intrusion prediction with machine learning**

**https://libres.uncg.edu/ir/uncg/f/S\_Suthaharan\_Big\_2014.pdf**

**ABSTRACT:** This paper focuses on the specific problem of Big Data classification of network intrusion traffic. It discusses the system challenges presented by the Big Data problems associated with network intrusion prediction. The prediction of a possible intrusion attack in a network requires continuous collection of traffic data and learning of their characteristics on the fly. The continuous collection of traffic data by the network leads to Big Data problems that are caused by the volume, variety and velocity properties of Big Data. The learning of the network characteristics requires machine learning techniques that capture global knowledge of the traffic patterns. The Big Data properties will lead to significant system challenges to implement machine learning frameworks. This paper discusses the problems and challenges in handling Big Data classification using geometric representation-learning techniques and the modern Big Data networking technologies. In particular this paper discusses the issues related to combining supervised learning techniques, representation-learning techniques, machine lifelong learning techniques and Big Data technologies (e.g. Hadoop, Hive and Cloud) for solving network traffic classification problems.

**3.3 Efficient Machine Learning for Big Data: A Review**

**https://www.researchgate.net/publication/273788293\_Efficient\_Machine\_Learning\_for\_Big\_Data\_A\_Review**

**ABSTRACT:** With the emerging technologies and all associated devices, it is predicted that massive amount of data will be created in the next few years, in fact, as much as 90% of current data were created in the last couple of years,a trend that will continue for the foreseeable future. Sustainable computing studies the process by which computer engineer/scientist designs computers and associated subsystems efficiently and effectively with minimal impact on the environment. However, current intelligent machine-learning systems are performance driven, the focus is on the predictive/classification accuracy, based on known properties learned from the training samples. For instance, most machine-learning-based nonparametric models are known to require high computational cost in order to find the global optima. With the learning task in a large dataset, the number of hidden nodes within the network will therefore increase significantly, which eventually leads to an exponential rise in computational complexity. This paper thus reviews the theoretical and experimental data-modeling literature, in large-scale data-intensive fields, relating to: (1) model efficiency, including computational requirements in learning, and data-intensive areas structure and design, and introduces (2) new algorithmic approaches with the least memory requirements and processing to minimize computational cost, while maintaining/improving its predictive/classification accuracy and stability.

**3.4 A survey of open source tools for machine learning with big data in the Hadoop ecosystem**

[**https://journalofbigdata.springeropen.com/articles/10.1186/s40537-015-0032-1**](https://journalofbigdata.springeropen.com/articles/10.1186/s40537-015-0032-1)

**ABSTRACT:** With an ever-increasing amount of options, the task of selecting machine learning tools for big data can be difficult. The available tools have advantages and drawbacks, and many have overlapping uses. The world’s data is growing rapidly, and traditional tools for machine learning are becoming insufficient as we move towards distributed and real-time processing. This paper is intended to aid the researcher or professional who understands machine learning but is inexperienced with big data. In order to evaluate tools, one should have a thorough understanding of what to look for. To that end, this paper provides a list of criteria for making selections along with an analysis of the advantages and drawbacks of each. We do this by starting from the beginning, and looking at what exactly the term “big data” means. From there, we go on to the Hadoop ecosystem for a look at many of the projects that are part of a typical machine learning architecture and an understanding of how everything might fit together. We discuss the advantages and disadvantages of three different processing paradigms along with a comparison of engines that implement them, including MapReduce, Spark, Flink, Storm, and H2O. We then look at machine learning libraries and frameworks including Mahout, MLlib, SAMOA, and evaluate them based on criteria such as scalability, ease of use, and extensibility. There is no single toolkit that truly embodies a one-size-fits-all solution, so this paper aims to help make decisions smoother by providing as much information as possible and quantifying what the tradeoffs will be. Additionally, throughout this paper, we review recent research in the field using these tools and talk about possible future directions for toolkit-based learning.

**3.5 Petuum: A New Platform for Distributed Machine Learning on Big Data**

<http://www.cs.cmu.edu/~seunghak/petuum_kdd15.pdf>

**ABSTRACT:** How can one build a distributed framework that allows efficient deployment of a wide spectrum of modern advanced machine learning (ML) programs for industrial-scale problems using Big Models (100s of billions of parameters) on Big Data (terabytes or petabytes)? Contemporary parallelization strategies employ fine-grained operations and scheduling beyond the classic bulk-synchronous processing paradigm popularized by MapReduce, or even specialized operators relying on graphical representations of ML programs. The variety of approaches tends to pull systems and algorithms design in different directions, and it remains difficult to find a universal platform applicable to a wide range of different ML programs at scale. We propose a general-purpose framework that systematically addresses data- and model-parallel challenges in large-scale ML, by leveraging several fundamental properties underlying ML programs that make them different from conventional operation-centric programs: error tolerance, dynamic structure, and nonuniform convergence; all stem from the optimization-centric nature shared in ML programs’ mathematical definitions, and the iterativeconvergent behavior of their algorithmic solutions. These properties present unique opportunities for an integrative system design, built on bounded-latency network synchronization and dynamic load-balancing scheduling, which is efficient, programmable, and enjoys provable correctness guarantees. We demonstrate how such a design in light of MLfirst principles leads to significant performance improvements versus well-known implementations of several ML programs, allowing them to run in much less time and at considerably larger model sizes, on modestly-sized computer clusters.

**SYSTEM ANALYSIS**

**4.SYSTEM ANALYSIS**

**4.1 EXISTING SYSTEM:**

Content relevant to viewpoints being expressed in a variety of domains, including the economics, society, and culture, is included in the data on SNS. Therefore, information on numerous flows and opinions on subjects like society, the economy, and politics can be derived by analysing the data on SNS. However, it is highly challenging and time-consuming to effectively analyse the data on SNS as it consists of a mixture of useful data, data from advertisements, and positive data that is helpful to the actual analysis. Studies on stable data collection and storage as well as effective data processing with constrained computing resources have been done recently as interest in big-data processing has grown. However, there is a dearth of studies and research on the use of big data before processing.

**4.1.1 DISADVANTAGES OF EXISTING SYSTEM:**

* 1. it is very difficult
  2. time-consuming

# 4.2 Proposed System:

Therefore, this study investigates how to effectively filter garbage data from big data, and thereby improve the accuracy and speed of the data analysis in real big-data processing. In particular, this study focuses on improving the filtering accuracy by including machine learning in the process of filtering garbage data. Therefore, in this study, we propose an algorithm that can improve the garbage data filtering accuracy of SNS big data by cyclic learning and prove the effectiveness of the algorithm through experiments.

# 4.2.1 Advantages of proposed system:

1. Effectively filter garbage data from big data

2. thereby improve the accuracy and speed of the data analysis

### **4.3 FUNCTIONAL REQUIREMENTS**

1.Data Collection

2.Data Preprocessing

3.Training And Testing

4.Modiling

5.Predicting

### **4.4 NON FUNCTIONAL REQUIREMENTS**

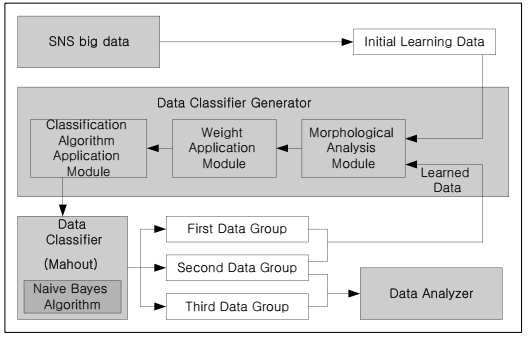
NON-FUNCTIONAL REQUIREMENT (NFR) specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software system. Example of nonfunctional requirement, *“how fast does the website load?”* Failing to meet non-functional requirements can result in systems that fail to satisfy user needs. Non- functional Requirements allows you to impose constraints or restrictions on the design of the system across the various agile backlogs. Example, the site should load in 3 seconds when the number of simultaneous users are > 10000. Description of non-functional requirements is just as critical as a functional requirement.

* Usability requirement
* Serviceability requirement
* Manageability requirement
* Recoverability requirement
* Security requirement
* Data Integrity requirement
* Capacity requirement
* Availability requirement
* Scalability requirement
* Interoperability requirement
* Reliability requirement
* Maintainability requirement
* Regulatory requirement
* Environmental requirement

**SYSTEM DESIGN**

**5. SYSTEM DESIGN**

**5.1 SYSTEM ARCHITECTURE:**

****

**Fig.5.1.1 System architecture**

**DATA FLOW DIAGRAM:**

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

**UPLOAD SNS DATASET**

**VERIFY**

**NO PROCESS**

**Yes NO**

**DATA CLASSIFIER GENERATOR**

**DATA CLASSIFIER USING SPARK NAÏVE BAYES**

**RUN EXTENSION RANDOM FOREST**

**RUN EXTENSION DECISION TREE**

**RUN EXTENSION XGBOOST**

**DATA ANALYZER**

**ACCURACY COMPARISON GRAPH**

**End process**

**Fig.5.1.3 Dataflow diagram**

**5.2 UML DIAGRAMS**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

**Use case diagram:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



**Fig.5.2.1 Usecase diagram**

**Class diagram:**

The class diagram is used to refine the use case diagram and define a detailed design of the system. The class diagram classifies the actors defined in the use case diagram into a set of interrelated classes. The relationship or association between the classes can be either an "is-a" or "has-a" relationship. Each class in the class diagram may be capable of providing certain functionalities. These functionalities provided by the class are termed "methods" of the class. Apart from this, each class may have certain "attributes" that uniquely identify the class.



**Fig.5.2.2 Class diagram**

**Activity diagram:**

The process flows in the system are captured in the activity diagram. Similar to a state diagram, an activity diagram also consists of activities, actions, transitions, initial and final states, and guard conditions.

Upload SNS dataset

Data classifier generator

Data classifier using SPARK naïve bayes

Run extension random forest

Run extension decision tree

Run extension XGBoost

Data analyzer

Accuracy comparison graph

**Fig.5.2.3 Activity diagram**

**Sequence diagram:**

A sequence diagram represents the interaction between different objects in the system. The important aspect of a sequence diagram is that it is time-ordered. This means that the exact sequence of the interactions between the objects is represented step by step. Different objects in the sequence diagram interact with each other by passing "messages".



**Fig.5.2.4 Sequence diagram**

**Collaboration diagram:**

A collaboration diagram groups together the interactions between different objects. The interactions are listed as numbered interactions that help to trace the sequence of the interactions. The collaboration diagram helps to identify all the possible interactions that each object has with other objects.

****

**Fig.5.2.5 Collaboration diagram**

**Component diagram:**

The component diagram represents the high-level parts that make up the system. This diagram depicts, at a high level, what components form part of the system and how they are interrelated. A component diagram depicts the components culled after the system has undergone the development or construction phase.



**Fig.5.2.6 Component diagram**

**Deployment diagram:**

The deployment diagram captures the configuration of the runtime elements of the application. This diagram is by far most useful when a system is built and ready to be deployed.

****

**Fig.5.2.7 Deployment diagram**

**IMPLEMENTATION**

**6. IMPLEMENTATION**

Now-a-days almost all peoples are using Social Networks Services to post their opinions on different topics such as Political opinion, online product opinion after purchasing and give opinions on many other topics. Often such TWEETS or POSTS contains irrelevant data called as Advertisement, Garbage (meaningless) and Definite (important or relevant post) and due to many users huge data will be collected (Big Data) which makes difficult in processing to extract relevant data and ignore garbage data.

To overcome from above issues author of this paper applying Big Data technologies such as HADOOP, SPARK or MAHOUT which can process data faster and help in extracting relevant data faster.

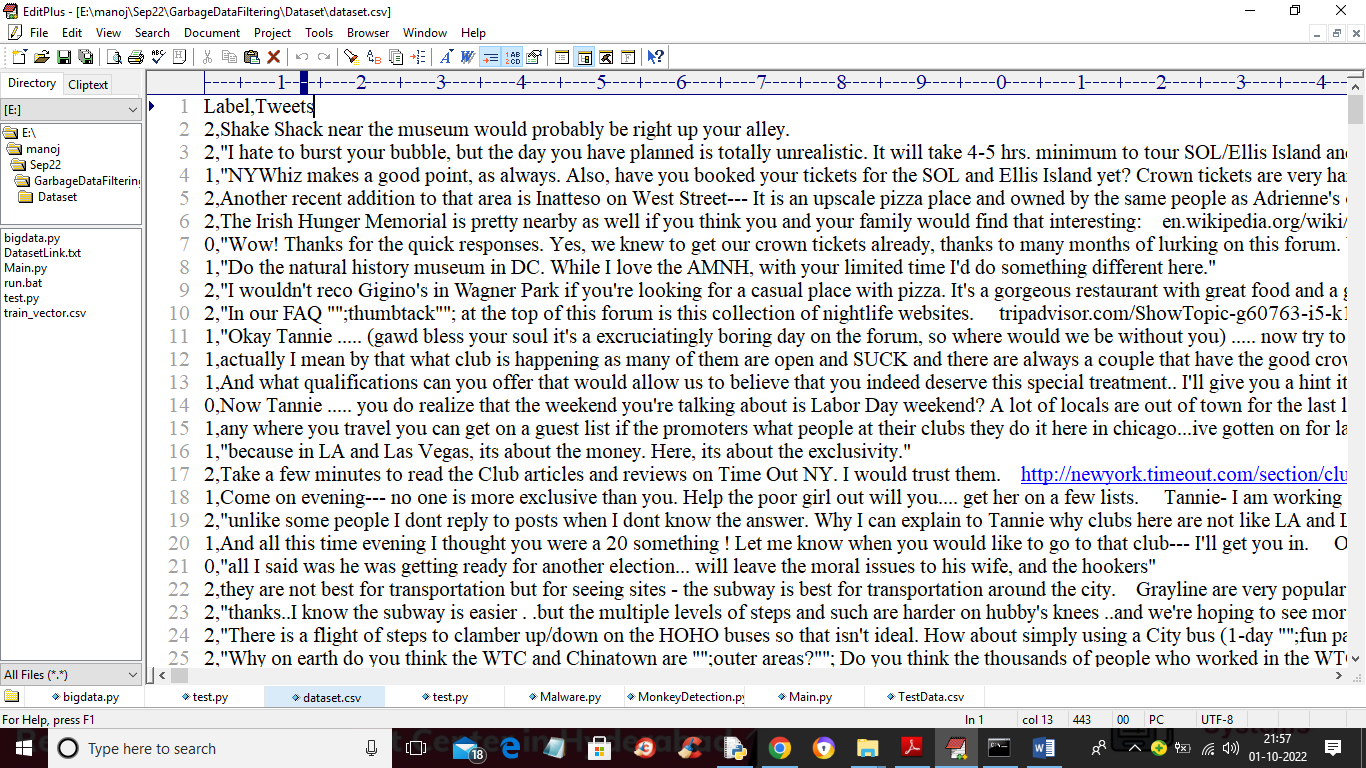
In same paper author is using Machine Learning algorithm called Naïve Bayes to classify POSTS or TWEETS in to different group called Garbage, Advertisement and Definite (relevant post).

Morphological weight (average occurrence of each work also called as weight) will be extracted from each post and this weight help machine learning to identify group of POST. If POST contains Garbage or Advertisement then same word may occur more number of times and this weight will get increase and if weight increase then POST will be consider as Garbage or Advertisement.

Machine learning Naïve Bayes algorithm will get trained on TWEETS data of different groups and then generate trained model. Whenever we applied TEST data on Trained Model then Naïve Bayes will calculate weight and based on weight value it will classify POST as Garbage or Advertisement or Definite.

Extension Concept: In propose paper author has used only one machine learning algorithm called Naïve Bayes and in extension we have decided to use Advance algorithm called Random Forest, XGBOOST and Decision Tree and this algorithms giving better accuracy compare to Naïve Bayes.

In propose paper author using some E-Commerce Tweets but not publish that dataset on internet so we downloaded own tweets on different categories. Below screen showing dataset details



In above screen first row contains dataset column names and remaining rows contains dataset values. First column contains label 0 (Garbage), 1 (advertisement) and 2 Definite. Second column contains TWEETS. So by using above dataset we will train SPARK Naïve Bayes and other algorithms.

MODULES:

To implement this project author has used following modules

1. Upload SNS Dataset: using this module we will upload Social Network Services dataset to application
2. Data Classifier Generator: using this module we will read all dataset tweets and then calculate weight of each words by using its occurrence in the Tweets.
3. Data Classifier using SPARK Naive Bayes: by using tweets weights we will train SPARK Naïve Bayes algorithm and then perform prediction on test data and then calculate its prediction accuracy.
4. Run Extension Random Forest: using this module we will train Extension Random Forest Algorithm and then perform prediction on test data and then calculate its prediction accuracy
5. Run Extension Decision Tree: using this module we will train Extension Decision Tree Algorithm and then perform prediction on test data and then calculate its prediction accuracy
6. Run Extension XGBOOST: using this module we will train Extension XGBOOST Algorithm and then perform prediction on test data and then calculate its prediction accuracy
7. Data Analyzer: using this module we will upload test data and then Trained Model will classify TWEETS into one of 3 groups called as 0 (Garbage), 1 (Advertisement) or 2 (definite)
8. Accuracy Comparison Graph: using this module we will plot accuracy comparison graph between all algorithms

Decision Tree: a tree diagram which is used for making decisions in business or computer programming and in which the branches represent choices with associated risks, costs, results, or probabilities.

Random Forest: Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

Naïve bayes: A naive Bayes classifier is an algorithm that uses Bayes' theorem to classify objects. Naive Bayes classifiers assume strong, or naive, independence between attributes of data points. Popular uses of naive Bayes classifiers include spam filters, text analysis and medical diagnosis.

XBOOST: XGBoost stands for Extreme Gradient Boosting. It uses more accurate approximations to find the best tree model. Boosting: N new training data sets are formed by random sampling with replacement from the original dataset, during which some observations may be repeated in each new training data set.

**6.2 SAMPLE CODE:**

*# Vector (Series) & Matrix (DateFrame) manipulation*

**import** numpy **as** np

**import** pandas **as** pd

*# Data Visualization*

**import** matplotlib.pyplot **as** plt

**from** mpl\_toolkits.mplot3d **import** Axes3D

**import** seaborn **as** sns

*# If JaveScript is configured and enabled:*

*# static images:*

*# %matplotlib inline*

*# interactive images:*

*# %matplotlib notebook*

*# Interactive Data Visualization*

*# import plotly.express as px*

*# Python Utilities*

*# Generate datetime objects from raw timestamps and vice versa*

**from** datetime **import** datetime

*# OS Interface*

*# import os*

*# Regex search patterns*

*# import re*

*# calling np.version.version should return 1.18.1*

*# np.version.version*

*# calling pd.\_\_version\_\_ should return 1.1.2*

*# pd.\_\_version\_\_*

*# pd.set\_option('display.max\_columns',None)*

*# avoid truncate view of DataFrame (scroll to view all columns); set to 0 for pandas to auto-detect the with of the terminal and print truncated object that fits the screen width*

*# pd.set\_option('float\_format', '{:.2f}'.format)*

*# prints floats with two decimal points; do not comment out in this project since the features lat and lon have sigficant figures after two decimal poin*

*# Display all outsputs if the cell has multiple commands as its input*

*# from IPython.core.interactiveshell import InteractiveShell*

*# InteractiveShell.ast\_node\_interactivity = "all"*

*# To ignore warnings*

*# import warnings*

*# warnings.filterwarnings('ignore')*

*# Split Function (see Signature for correct tuple unpacking)*

*# from sklearn.model\_selection import train\_test\_split*

*# Default split*

*# X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.33, random\_state=42)*

*# When perfroming a classic Train | Test Spit fit ONLY to X\_train to avoid data leakage! (Follow Procedure described in documentation under Cross Validation and Linear Regression Project)*

*# Data Scaling (iff values are in different order of magnitude)*

*# from sklearn.preprocessing import StandardScaler*

*# k-fold cross validation scores; estimator = ML model, cv = fold value, scoring = error metric (use the ones provided by sklearn!)*

*# from sklearn.model\_selection import cross\_val\_score*

*# Polynomial Regression ( poly\_trafo: X->X\*...\*X )*

*# from sklearn.preprocessing import PolynomialFeatures*

*# Grid search with cross vadlidation*

*# from sklearn.model\_selection import GridSearchCV*

*# Linear Regression Model*

*# from sklearn.linear\_model import LinearRegression*

*# Elastic Net Regularization: start here for regularization in Linear Regression. Make sure to keep an l1\_ratio that allows us to go fully to Lasso or fully to Ridge. See Lasso and Ridge explanations below.*

*# from sklearn.linear\_model import ElasticNetCV*

*# Use from sklearn.linear\_model import ElasticNet in case CV done manually / grid search*

*# Standard procedure with no grid search: create X and y, split data, scale data (stadardize)*

*# Standard procedure with grid search: create X and y, split data, scale data (stadardize), instantiate base model,*

*# L2: Ridge Regularization: adds beta squared shrinkage penalty. Hyper-parameter alpha: alpha=0 -> RSS minimization. L2 CV takes an alpha tuple and computes the hyper-parameter that delivers the best performance (either based on default scorer or one from the SCORES dictionary)*

*# from sklearn.linear\_model import RidgeCV*

*# Use from sklearn.linear\_model import Ridge in case CV done manually*

*# L1: Lasso Regularization: adds absolute beta value shrinkage penalty. Hyper-parameter alpha: alpha=0 -> RSS minimization. There are two ways to determine the alpha hyper-parameter: (a) provide list of alphas as an array (b) alpha can be set automatically by the class based off epsilon and n\_alphas (we use the default values)*

*# from sklearn.linear\_model import LassoCV*

*# Use from sklearn.linear\_model import Lasso in case no CV done manually*

**SOFTWARE ENVIRONMENT**

**7.SOFTWARE ENVIRONMENT**

# What is Python :-

Below are some facts about Python.

Python is currently the most widely used multi-purpose, high-level programming language.

Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.

Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.

Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

The biggest strength of Python is huge collection of standard library which can be used for the following –

* + [Machine Learning](https://www.geeksforgeeks.org/machine-learning/)
  + GUI Applications (like Kivy, Tkinter, PyQt etc. )
  + Web frameworks like Django (used by YouTube, Instagram, Dropbox)
  + Image processing (like Opencv, Pillow)
  + Web scraping (like Scrapy, BeautifulSoup, Selenium)
  + Test frameworks
  + Multimedia

### Advantages of Python :-

Let’s see how Python dominates over other languages.

#### 1. Extensive Libraries

Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don’t have to write the complete code for that manually.

#### 2. Extensible

As we have seen earlier, Python can be**extended to other languages**. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

#### 3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add **scripting capabilities**to our code in the other language.

#### 4. Improved Productivity

The language’s simplicity and extensive libraries render programmers**more productive** than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

#### 5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet Of Things. This is a way to connect the language with the real world.

#### 6. Simple and Easy

When working with Java, you may have to create a class to print **‘Hello World’**. But in Python, just a print statement will do. It is also quite **easy to learn, understand,** and**code.** This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

#### 7. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and **indentation is mandatory.** This further aids the readability of the code.

#### 8. Object-Oriented

This language supports both the **procedural and object-oriented**programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the **encapsulation of data** and functions into one.

#### 9. Free and Open-Source

Like we said earlier, Python is **freely available.** But not only can you[**download Python**](https://data-flair.training/blogs/install-python-windows/) for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

#### 10. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn’t the same with Python. Here, you need to**code only once**, and you can run it anywhere. This is called **Write Once Run Anywhere (WORA)**. However, you need to be careful enough not to include any system-dependent features.

#### 11. Interpreted

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, **debugging is easier** than in compiled languages.

Any doubts till now in the advantages of Python? Mention in the comment section.

### **Advantages of Python Over Other Languages**

#### 1. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don’t have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

#### 2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

**The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.**

#### 3. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and [**machine learning**](https://data-flair.training/blogs/machine-learning-tutorials-home/), automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

### **Disadvantages of Python**

So far, we’ve seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let’s now see the downsides of choosing Python over another language.

#### 1. Speed Limitations

We have seen that Python code is executed line by line. But since [Python](https://www.python.org/) is interpreted, it often results in **slow execution**. This, however, isn’t a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

#### 2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the **client-side**. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called **Carbonnelle**.

The reason it is not so famous despite the existence of Brython is that it isn’t that secure.

#### 3. Design Restrictions

As you know, Python is **dynamically-typed**. This means that you don’t need to declare the type of variable while writing the code. It uses **duck-typing**. But wait, what’s that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can**raise run-time errors**.

#### 4. Underdeveloped Database Access Layers

Compared to more widely used technologies like **JDBC (Java DataBase Connectivity)** and **ODBC (Open DataBase Connectivity)**, Python’s database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

#### 5. Simple

No, we’re not kidding. Python’s simplicity can indeed be a problem. Take my example. I don’t do Java, I’m more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

**History of Python : -**

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde &Informatica). The greatest achievement of ABC was to influence the design of Python.Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners1, Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it."Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

**Python Development Steps : -**

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt.sources in February 1991. This release included already exception handling, functions, and the core data types of list, dict, str and others. It was also object oriented and had a module system.  
Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce, which Guido Van Rossum never liked.Six and a half years later in October 2000, Python 2.0 was introduced. This release included list comprehensions, a full garbage collector and it was supporting unicode.Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backwards compatible with Python 2.x. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it."Some changes in Python 7.3:

* Print is now a function
* Views and iterators instead of lists
* The rules for ordering comparisons have been simplified. E.g. a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
* There is only one integer type left, i.e. int. long is int as well.
* The division of two integers returns a float instead of an integer. "//" can be used to have the "old" behaviour.
* Text Vs. Data Instead Of Unicode Vs. 8-bit

**Purpose :-**

We demonstrated that our approach enables successful segmentation of intra-retinal layers—even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

**Python**

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

* Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
* Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

**Modules Used in Project :-**

**Tensorflow**

TensorFlow is a [free](https://en.wikipedia.org/wiki/Free_software) and [open-source](https://en.wikipedia.org/wiki/Open-source_software) [software library for dataflow and differentiable programming](https://en.wikipedia.org/wiki/Library_(computing)) across a range of tasks. It is a symbolic math library, and is also used for [machine learning](https://en.wikipedia.org/wiki/Machine_learning) applications such as [neural networks](https://en.wikipedia.org/wiki/Neural_networks). It is used for both research and production at [Google](https://en.wikipedia.org/wiki/Google).‍

TensorFlow was developed by the [Google Brain](https://en.wikipedia.org/wiki/Google_Brain) team for internal Google use. It was released under the [Apache 2.0](https://en.wikipedia.org/wiki/Apache_License) [open-source license](https://en.wikipedia.org/wiki/Open-source_license) on November 9, 2015.

**Numpy**

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

* A powerful N-dimensional array object
* Sophisticated (broadcasting) functions
* Tools for integrating C/C++ and Fortran code
* Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

**Pandas**

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

**Matplotlib**

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and [IPython](http://ipython.org/) shells, the [Jupyter](http://jupyter.org/) Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the [sample plots](https://matplotlib.org/tutorials/introductory/sample_plots.html) and [thumbnail gallery](https://matplotlib.org/gallery/index.html).

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

**Scikit – learn**

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. **Python**

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

* Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
* Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

**Install Python Step-by-Step in Windows and Mac :**

Python a versatile programming language doesn’t come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

## How to Install Python on Windows and Mac :

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

**Note:** The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your **System Requirements**. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a **Windows 64-bit operating system**. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. [Download the Python Cheatsheet here.](https://myelearninghub.com/python-cheat-sheet/)The steps on how to install Python on Windows 10, 8 and 7 are **divided into 4 parts** to help understand better.

### Download the Correct version into the system

**Step 1:** Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: [**https://www.python.org**](https://www.python.org/)



Now, check for the latest and the correct version for your operating system.

**Step 2:** Click on the Download Tab.

****

**Step 3:** You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4

****

**Step 4:** Scroll down the page until you find the Files option.

**Step 5:** Here you see a different version of python along with the operating system.



• To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.

•To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

**Note:** To know the changes or updates that are made in the version you can click on the Release Note Option.

### Installation of Python

**Step 1:** Go to Download and Open the downloaded python version to carry out the installation process.



**Step 2:** Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.



**Step 3:** Click on Install NOW After the installation is successful. Click on Close.



With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

**Note:** The installation process might take a couple of minutes.

### Verify the Python Installation

**Step 1:** Click on Start

**Step 2:** In the Windows Run Command, type “cmd”.



**Step 3:** Open the Command prompt option.

**Step 4:** Let us test whether the python is correctly installed. Type **python –V** and press Enter.



**Step 5:** You will get the answer as 3.7.4

**Note:** If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

### Check how the Python IDLE works

**Step 1:** Click on Start

**Step 2:** In the Windows Run command, type “python idle”.



**Step 3:** Click on IDLE (Python 3.7 64-bit) and launch the program

**Step 4:** To go ahead with working in IDLE you must first save the file. **Click on File > Click on Save**



**Step 5:** Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

**Step 6:** Now for e.g. **enter print**.

**SYSTEM TESTING**

**8.SYSTEM TESTING**

**8.1 TESTING STRATEGIES**

**UNIT TESTING**

Unit testing, a testing technique using which individual modules are tested to determine if there are issues by the developer himself.. it is concerned with functional correctness of the standalone modules. The main aim is to isolate each unit of the system to identify, analyze and fix the defects.

Unit Testing Techniques:

Black Box Testing - Using which the user interface, input and output are tested.

White Box Testing –Used to test each one of those functions behavior is tested.

**DATA FLOW TESTING**

Data flow testing is a family of testing strategies based on selecting paths through the program’s control flow in order to explore sequence of events related to the status of Variables or data object. Dataflow Testing focuses on the points at which variables receive and the points at which these values are used.

**INTEGRATION TESTING**

Integration Testing done upon completion of unit testing, the units or modules are to be integrated which gives raise too integration testing. The purpose of integration testing is to verify the functional, performance, and reliability between the modules that are integrated.

**BIG BANG INTEGRATION TESTING**

Big Bang Integration Testing is an integration testing Strategy wherein all units are linked at once, resulting in a complete system. When this type of testing strategy is adopted, it is difficult to isolate any errors found, because attention is not paid to verifying the interfaces across individual units.

**USER INTERFACE TESTING**

User interface testing, a testing technique used to identify the presence of defects is a product/software under test by Graphical User interface [GUI].

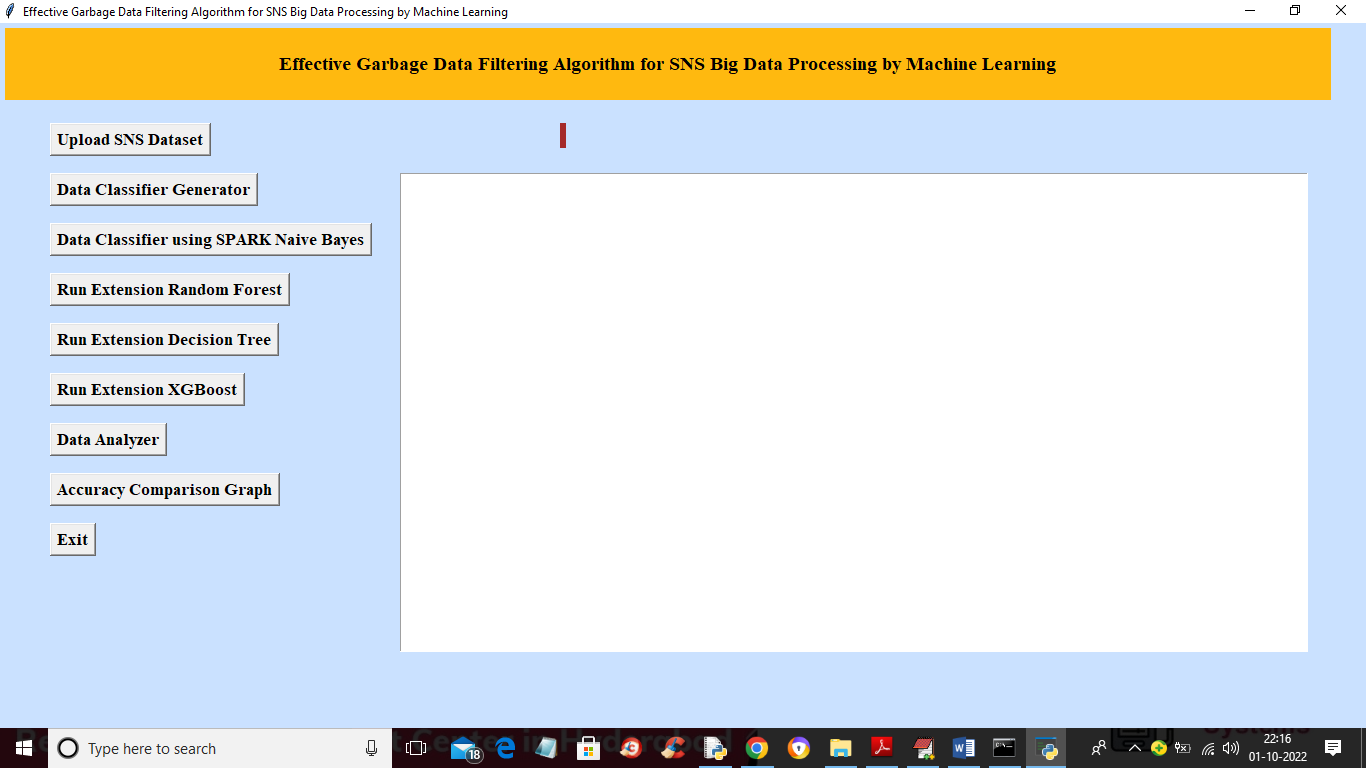
**8.2 TEST CASES:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **INPUT** | **If available** | **If not available** |
| 1 | Upload SNS Dataset | we will upload Social Network Services dataset to application | There is no process |
| 2 | Data Classifier Generator | we will read all dataset tweets and then calculate weight of each words | There is no process |
| 3 | Data Classifier using SPARK Naive Bayes | we will train SPARK Naïve Bayes algorithm | There is no process |
| 4 | Run Extension Random Forest | we will train Extension Random Forest Algorithm and then perform prediction | There is no process |
| 5 | Run Extension Decision Tree | we will train Extension Decision Tree Algorithm and then perform prediction | There is no process |
| 6 | Run Extension XGBOOST | we will train Extension XGBOOST Algorithm and then perform prediction | There is no process |
| 7 | Data Analyzer | we will upload test data and then Trained Model will classify TWEETS | There is no process |
| 8 | Accuracy Comparison Graph | we will plot accuracy comparison graph between all algorithms | There is no process |

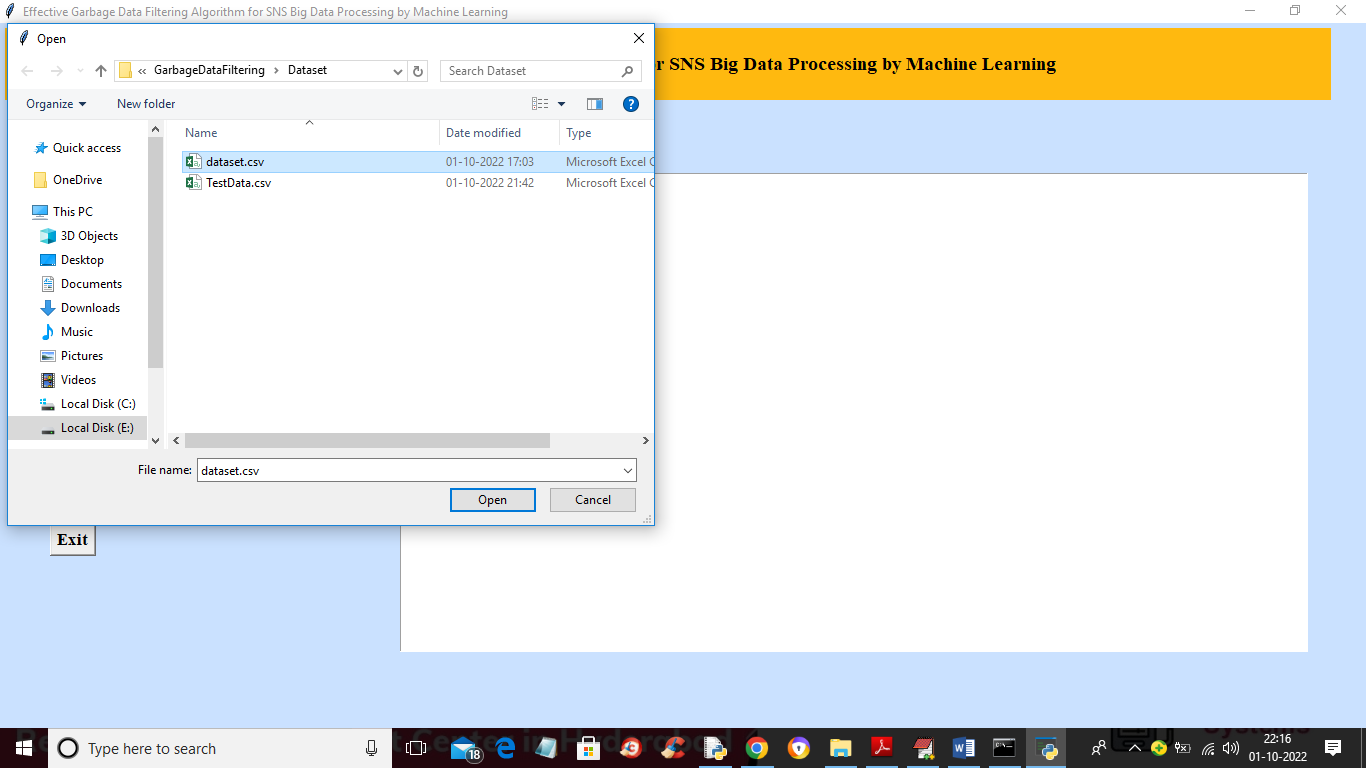
**SCREENS**

1. **SCREENSHOTS**

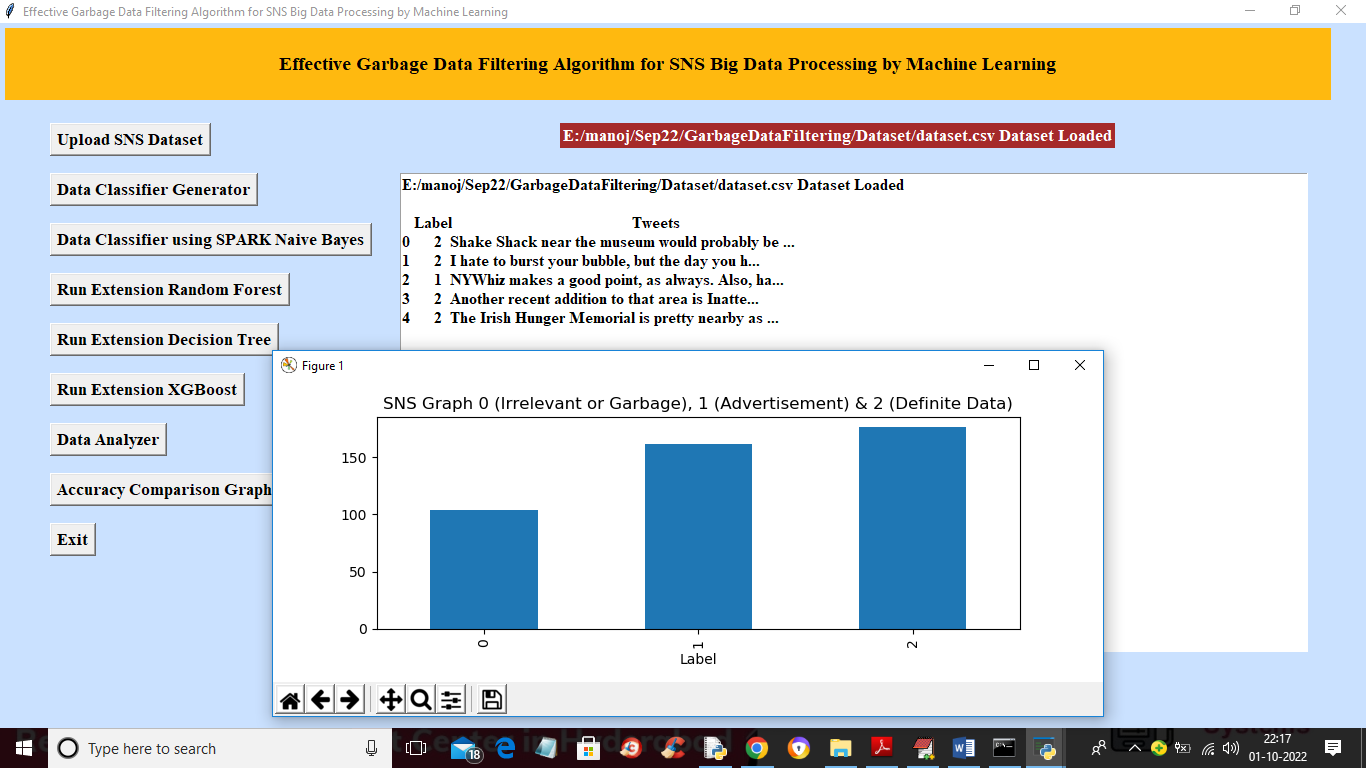
To run project double click on ‘run.bat’ file to get below screen



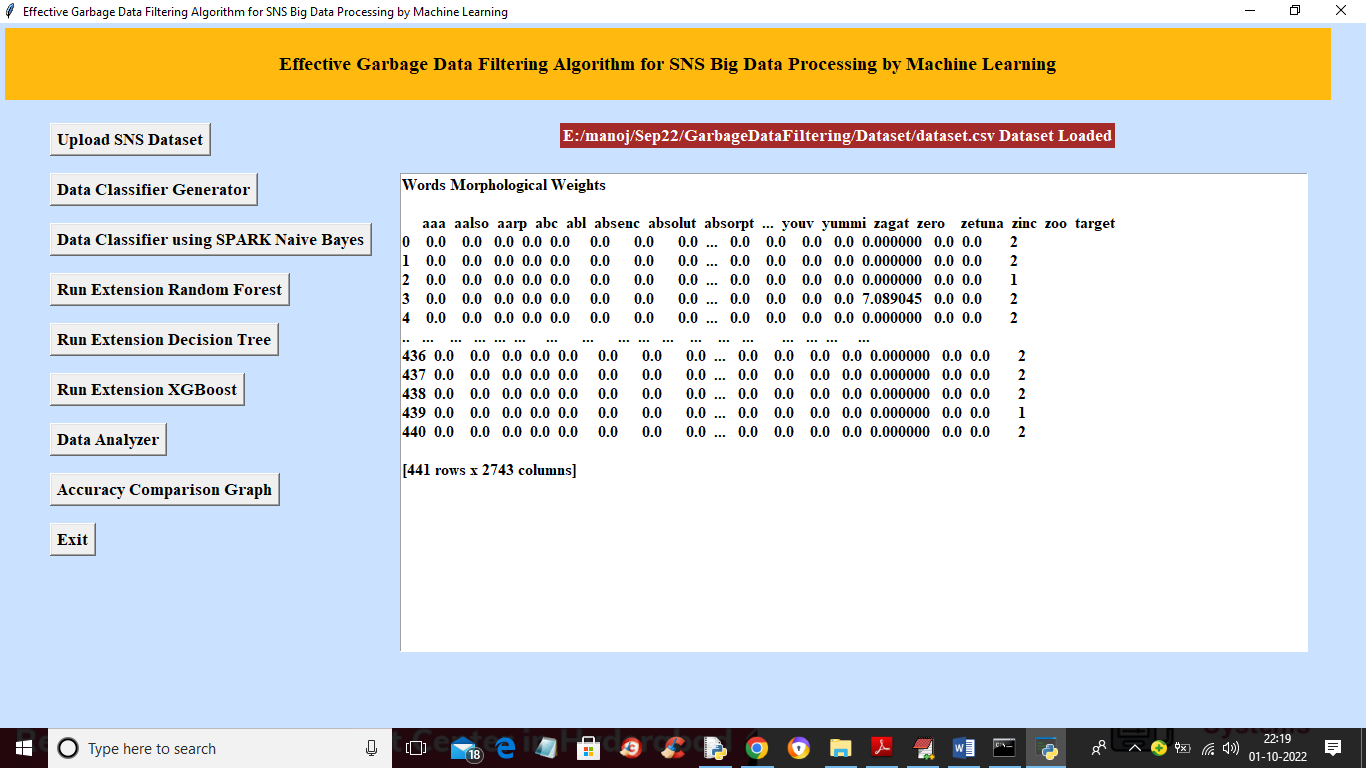
In above screen click on ‘Upload SNS Dataset’ button to load dataset and get below screen



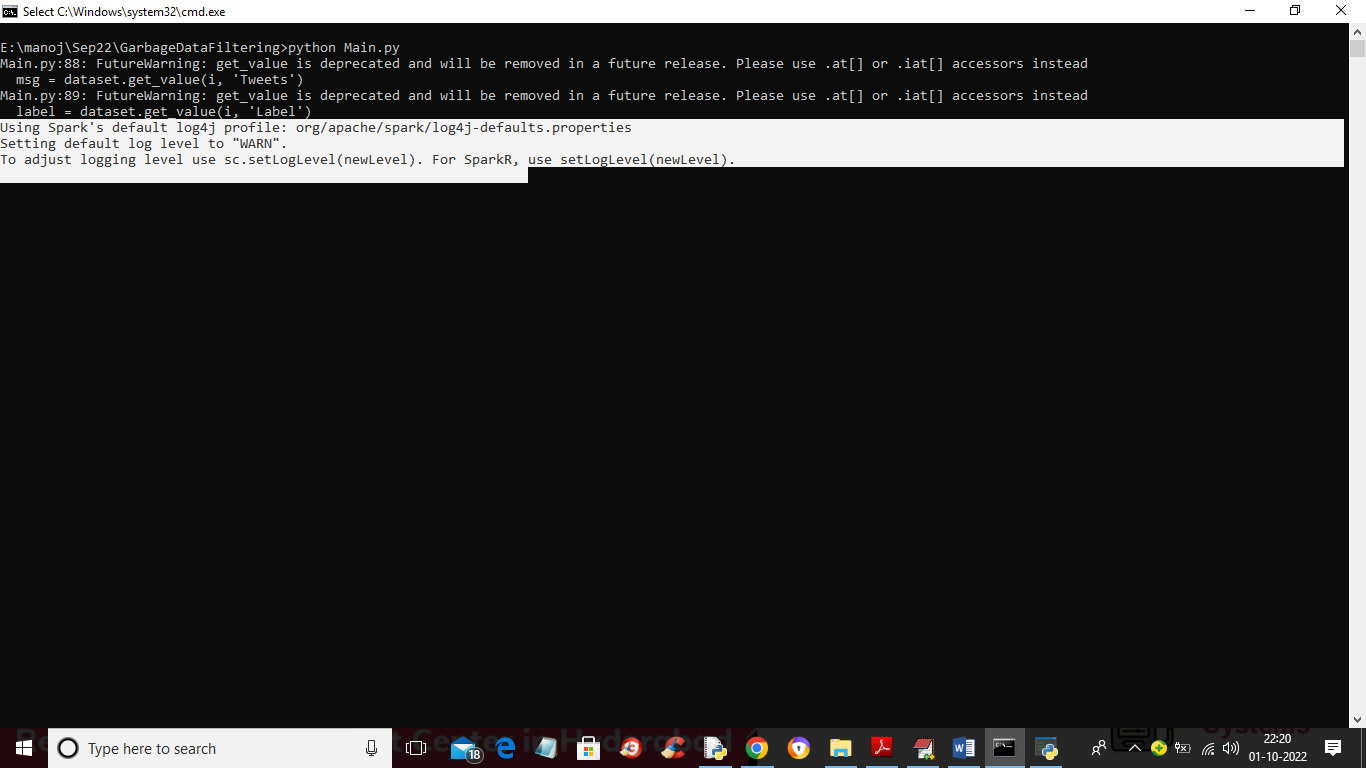
In above screen selecting and uploading ‘dataset.csv’ file and then click on ‘Open’ button to load dataset and get below output



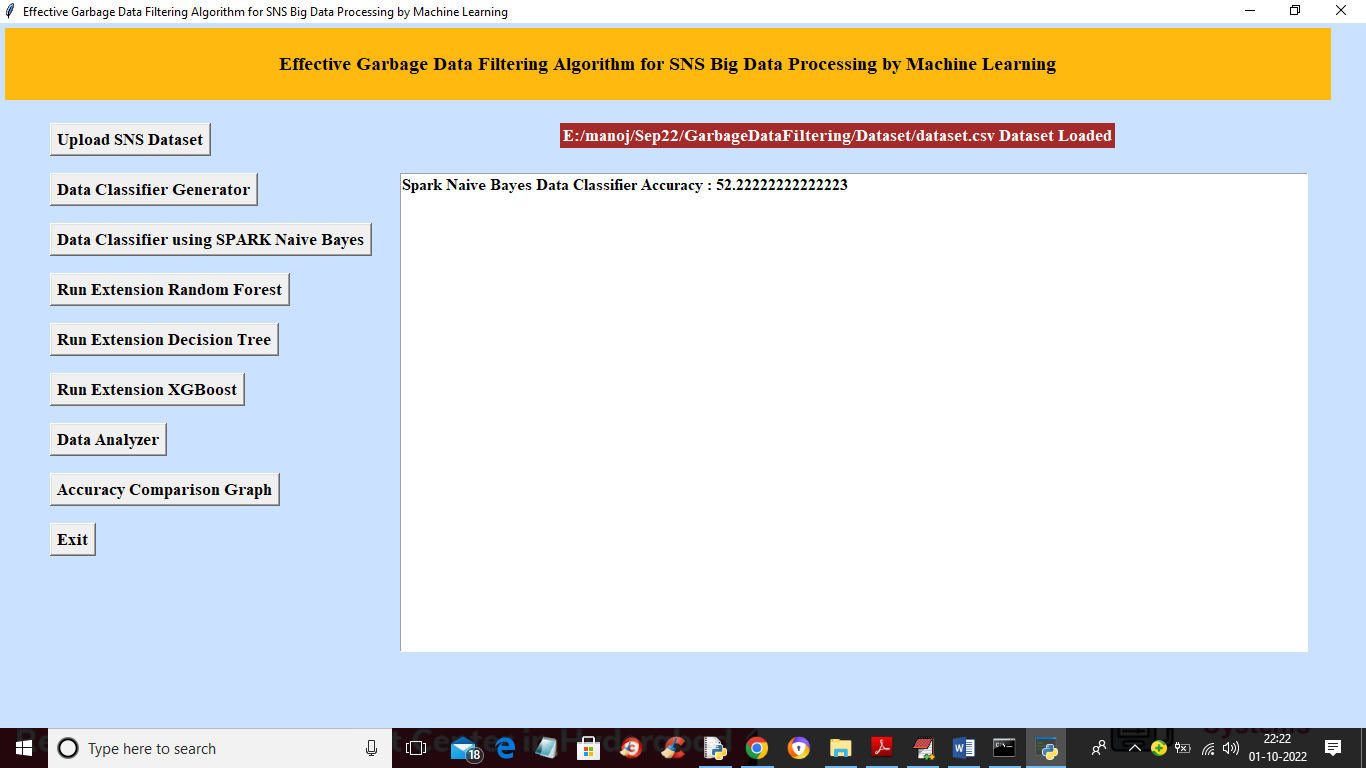
In above screen dataset loaded and in graph x-axis represents types of data as 0, 1 or 2 and y-axis represents number of records found in dataset in that group and now click on ‘Dataset Classifier Generator’ to convert dataset tweets into morphologic weights and get below output.



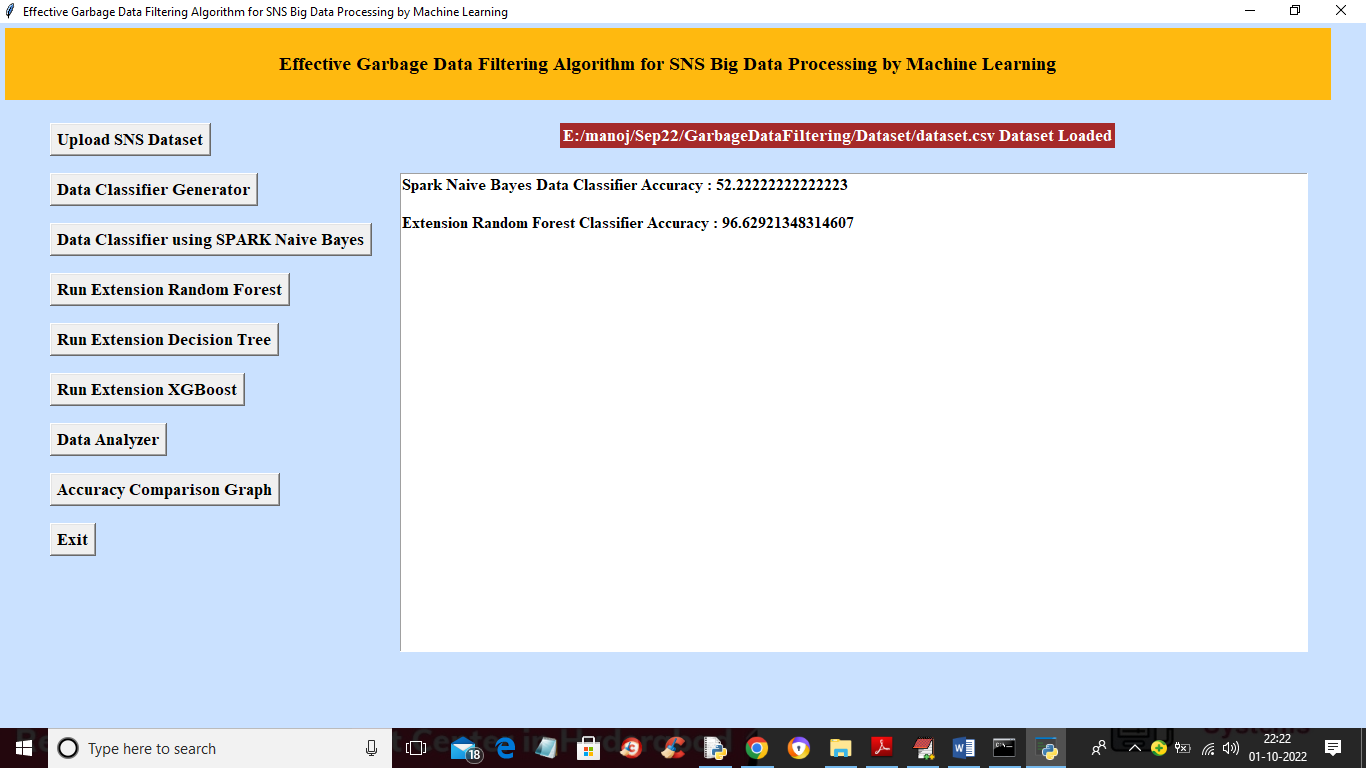
In above screen first row represents word and remaining rows contains weight of that word and now click on ‘Data Classifier using SPARK Naive Bayes’ button to train Naïve Bayes algorithm and get below prediction accuracy.



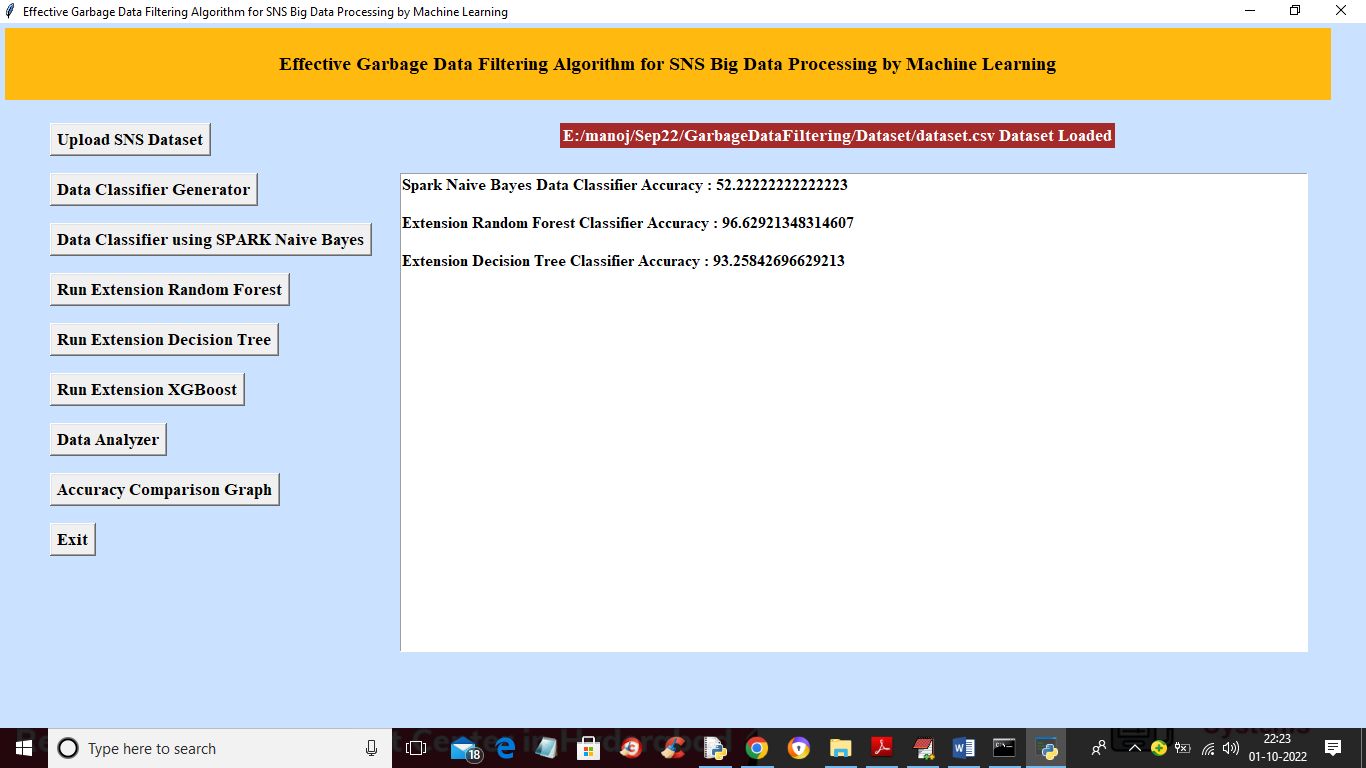
In above screen SPARK processing and naïve Bayes training started and after some time will get below output



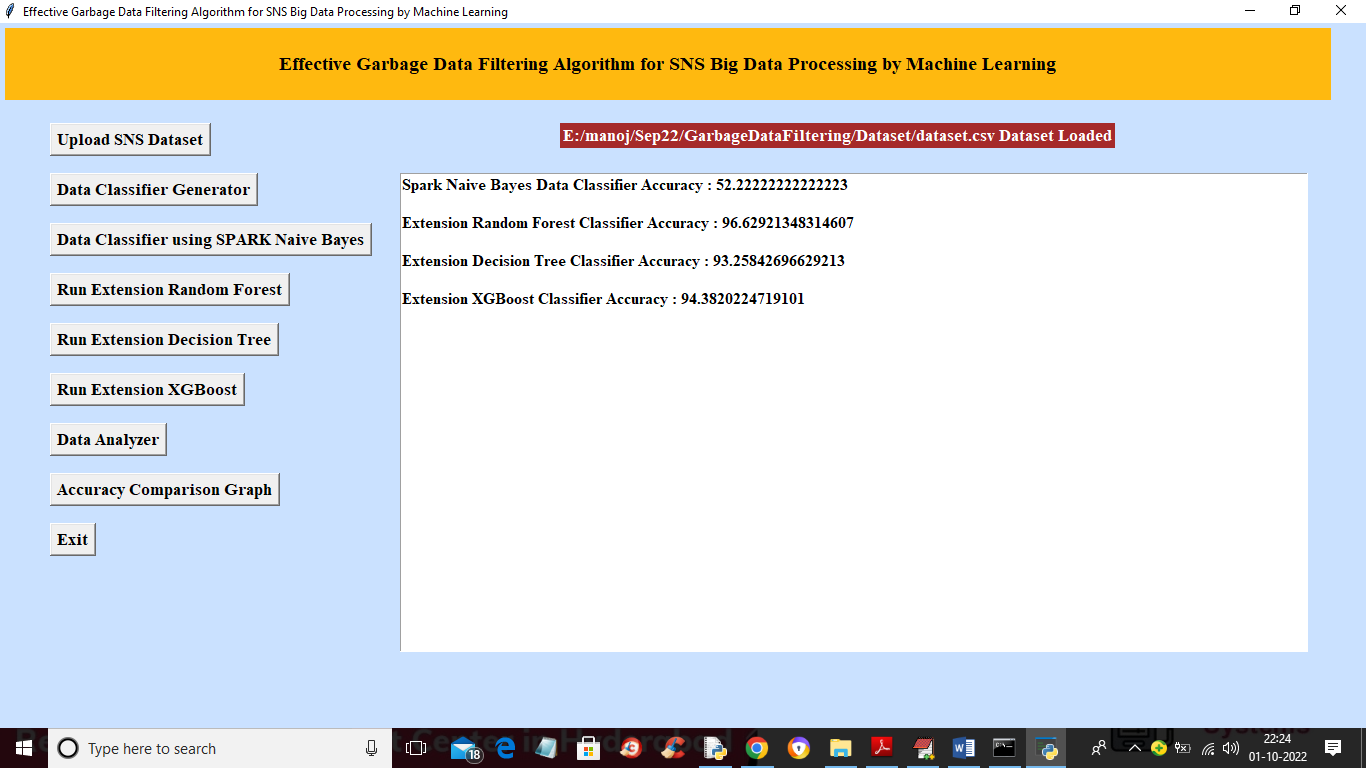
In above screen with Naïve Bayes we got 52% accuracy and now click on ‘Run Extension Random Forest’ button to train Random Forest and get below accuracy



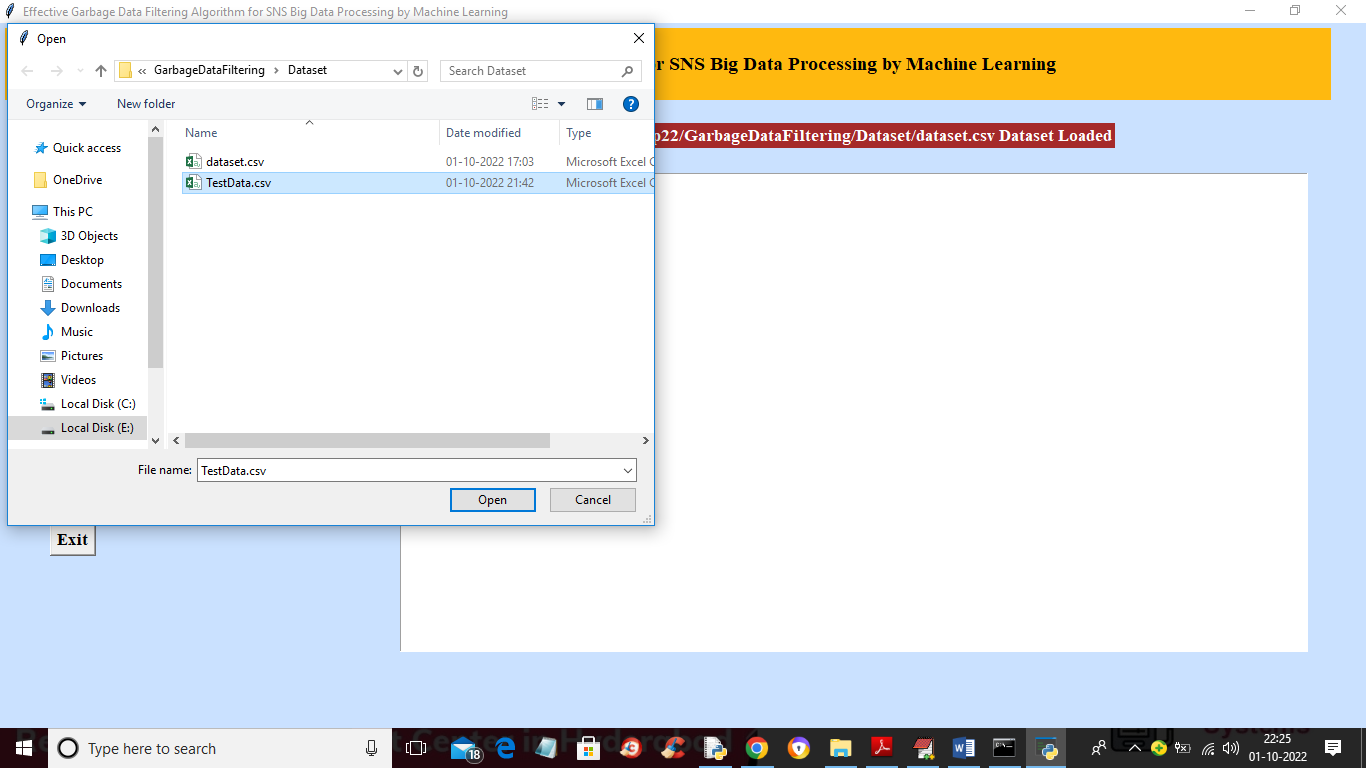
In above screen with Random Forest we got 96% accuracy and now click on ‘Run Extension Decision Tree’ button to train decision tree and get below accuracy



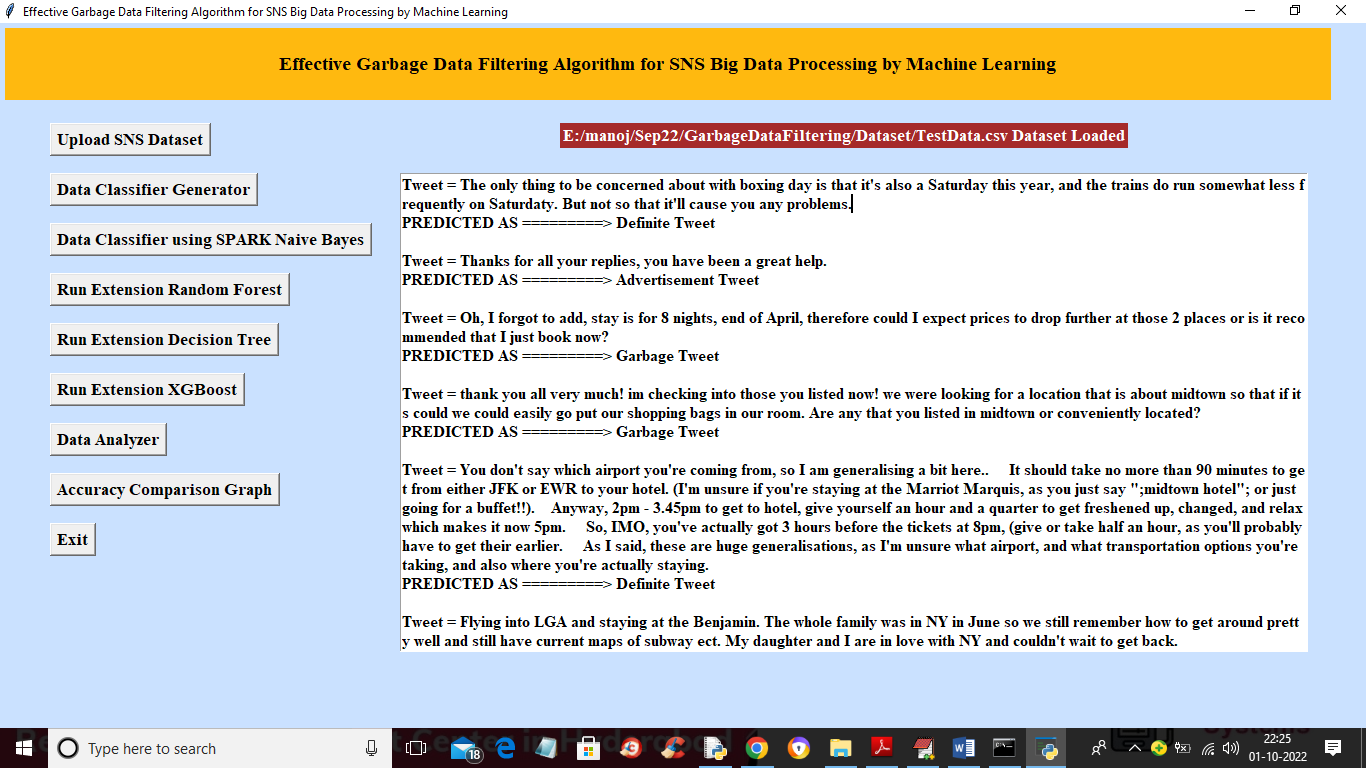
In above screen with Decision Tree we got 93% accuracy and now click on ‘Run Extension XGBoost’ button to train XGBOOST and get below accuracy



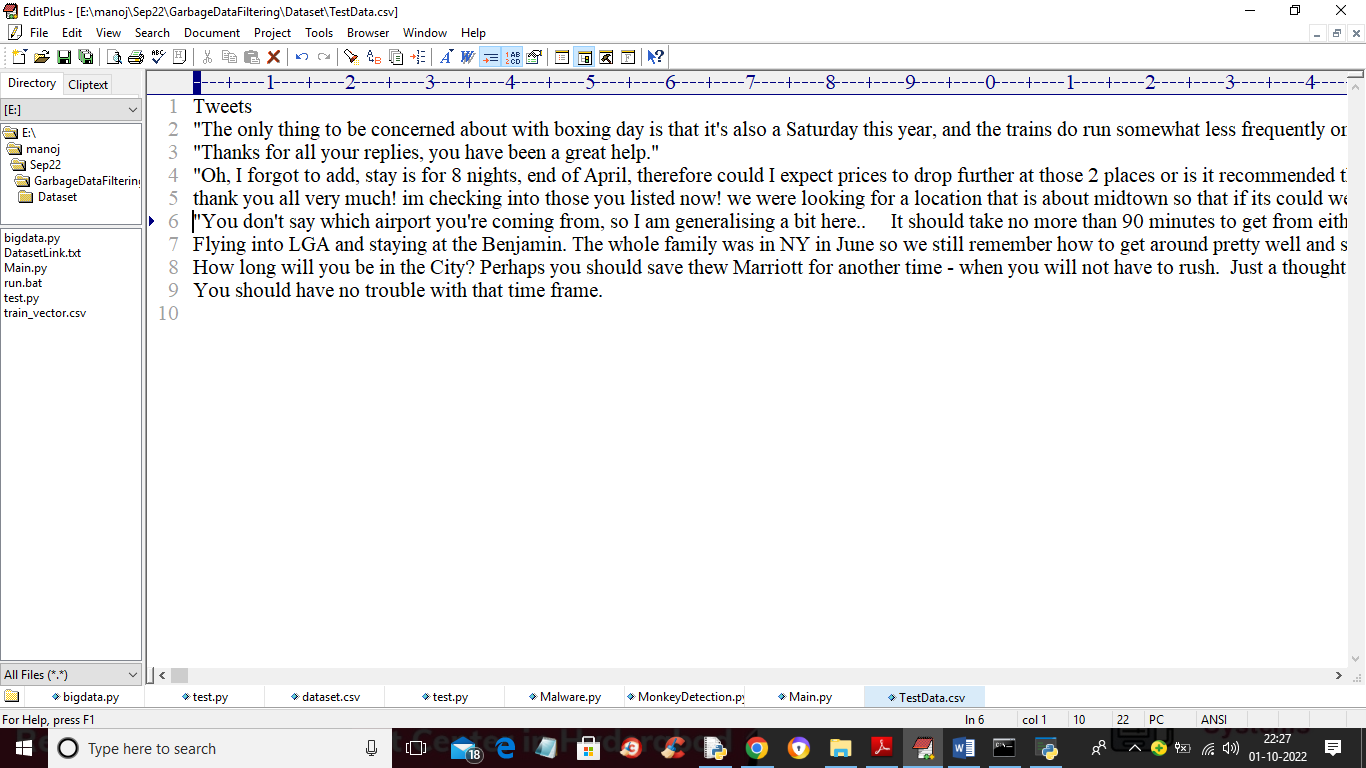
In above screen with XGBOOST we got 94% accuracy and now click on ‘Data Analyzer’ button to upload test data and then classifier algorithm will predict group of test data



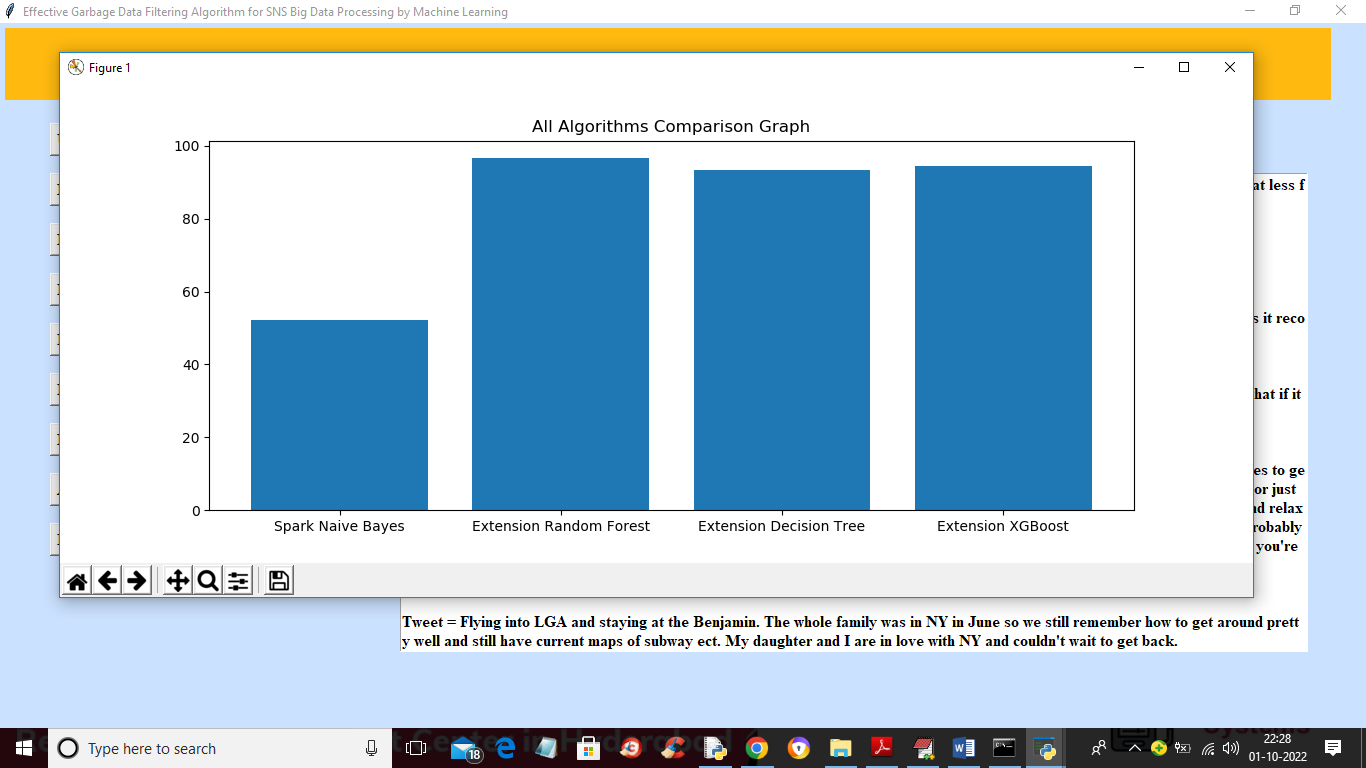
In above screen selecting and uploading ‘TestData.csv’ file and then click on ‘Open’ button to get below prediction output



In above screen after = (equal to) symbol we can see the TWEET and in next line after =🡺 arrow symbol we can see then prediction or classification result as Garbage, advertisement or Definite. In below screen of TestData.csv we can see it contains only tweets and Machine Learning algorithm will predict its group



In above test data we have only TWEETS and in prediction screen we got GROUP prediction from ML algorithms. Now click on ‘Accuracy Comparison Graph’ button to get below graph



In above graph x-axis represents algorithm names and y-axis represents accuracy of those algorithms and in above graph we can see all extension algorithms got high accuracy compare to propose algorithms

**CONCLUSION**

**10.CONCLUSION**

In this paper, we proposed and implemented an effective SNS garbage data filtering system through repetitive machine learning. We assume that the proposed system can improve the accuracy of the analysis of unstructured data in SNS by separating it into garbage, advertisement, and definite data through machine learning. Concerning the accuracy experiment, data filtering showed an accuracy of up to 74.45% following a comparison with the correct answer set. Therefore, it is found that it may be advantageous in a big data processing environment where a large amount of data must be processed quickly. Based on this, the contribution of this study is summarized as follows. First, this study proposed an effective garbage and advertisement data filtering system that can be used in big data processing system. It is designed to enhance the efficiency of big data processing by selecting and processing only data that is worth processing from a large amount of data generated in daily life such as SNS big data. Second, we introduced a recursive machine learning method for data filtering. We made initial learning data from SNS big data and used it for data filtering, and we improved the accuracy of filtering by using the filtered data as learning data through the proposed system. We think that the results of this study can be applied to the field of efficient processing of SNS Big Data and it can be used very well in various fields of obtaining valuable information from SNS Big Data.

**BIBILOGRAPHY**

**11. REFERENCES**

[1] J. Qiu, Q. Wu, G. Ding, Y. Xu, S. Feng, “A survey of machine learning for big data processing,” EURASIP J. Adv. Signal Process. vol. 2016, pp. 1-16, 2016.

[2] S. Suthanharan, “Big data classification: Problems and challenges in network intrusion prediction with machine learning,” ACM SIGMETRICS Perf. Eval. Rev. vol. 41, pp. 70-73, 2014.

[3] O. Jarrah, P. Yoo, S. Muhaidat, G. Karagiannidis, K. Taha, “Efficient Machine Learning for Big Data: A Review,” Big Data Res. vol. 2, pp. 87-93, 2015.

[4] S. Landset, T. Khoshgoftaar, A. Richter, T. Hasanin, “A survey of open source tools for machine learning with big data in the Hadoop ecosystem,” J. Big Data, vol. 2, pp. 1-36, 2015.

[5] E. Xing, Q. Ho, W. Dai, J. Kim, Y. Yu, “Petuum: A New Platform for Distributed Machine Learning on Big Data,” IEEE Trans. Big Data, vol. 1, pp. 49-67, 2015.

[6] M. Chen, Y. Hao, K. Hwang, L. Wang, L. Wang, “Disease Prediction by Machine Learning Over Big Data From Healthcare Communities,” IEEE Access, vol. 5, pp. 8869–8879, 2017.

[7] M. Gunasekaran, V. Vijayakumar, R. Varatharajan, K. Priyan S. Revathi, H. Ching-Hsien, “Machine Learning Based Big Data Processing Framework for Cancer Diagnosis Using Hidden Markov Model and GM Clustering,” Wireless Personal Communications, vol. 102, pp. 2099-2116, 2018.

[8] W. Xiaofei, Z. Yuhua, L. Victor, G. Nadra, J. Tianpeng, “D2D Big Data: Content Deliveries over Wireless Device-to-Device Sharing in Large-Scale Mobile Networks,” IEEE Wireless Communications. vol. 25, pp. 32-38, 2018.

[9] Z. Zhenhua, H. Qing, G. Jing, N. Ming, “A deep learning approach for detecting traffic accidents from social media data,” Transportation Research Part C: Emerging Technologies, vol. 86, pp. 580-596, 2017.

[10] S. Ou; J. Lee, “Implementation of a Spam Message Filtering System using Sentence Similarity Measurements,” KIISE Trans. Comput. Pract. (KTCP), vol. 23, pp. 57-64, 2017.

[11] D. Cho; K. Lim; S. Cho; S. Han; Y. Hwang, “Classifying Windows Executables using API-based Information and Machine Learning,” J. KIISE, vol. 43, pp. 1325-1333, 2016.

[12] H. Choi, J. Park, “Security tendency analysis techniques through machine learning algorithms applications in big data environments,” J. Digit. Converg. vol. 13, pp. 269-267, 2015.

[13] S. Jun, “A Big Data Preprocessing using Statistical Text Mining. J. Korean Inst,” Intell. Syst. vol. 25, pp. 470-476, 2015.

[14] M. Yang, M. Kiang, W. Shang, “Filtering big data from social media – Building an early warning system for adverse drug reactions,” J. Biomed. Inf. vol. 54, pp. 230-240, 2015.

[15] R. Hu, W. Dou, J. Liu, “ClubCF: A Clustering-Based Collaborative Filtering Approach for Big Data Application,” IEEE Trans. Emerg. Topics Comput. vol. 2, pp. 302-313, 2013.