**Group 5**

**Project Proposal Report**

**(****Effective Garbage Data Filtering)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **NAME OF THE STUDENT** | **E-Mail** | **R ID** |  | **Contributions** |
| Peela Hema Harsha Vardhan | hpeela@ttu.edu | R11912124 | ●  ● | Training the Data Model based on the machine learning algorithm. Evaluating the model based on its |
|  |  |  |
| Nitin Ancha | niancha@ttu.edu | R11892349 | ● | performance  Analysed Decision Tree  Algorithm |
| Kandula Thanmayee | tkandula@ttu.edu | R11931869 | ● | Gathering the requirements needed to develop the project. |
|  |  |  | ●  ● | Analysis to accept and achieve the  quality of data  Analysed Naive Bayes  Algorithm (Existing Work) |
| Kalyani Chittipolu | kchittip@ttu.edu | R11909106 |
| Sai Nymisha Issai | sissai@ttu.edu | R11892336 | ● | Training the Data Model based on the machine learning algorithm. |
|  |  |  | ●  ● | Evaluating the model based on its  performance  Analysed Random Forest  Algorithm |
| Nithish Chandra Vyas Talluru | ntalluru@ttu.edu | R11842825 |
| Sowmya Raavi | soraavi@ttu.edu | R11909388 | ● | Training the Data Model based on the machine learning algorithm. |
|  |  |  |
| Snehitha Pothina | spothina@ttu.edu | R11889528 | ●  ● | Evaluating the model based on its performance.  Analysed XG Boost Algorithm |

**● Problem Statement:**

In the era of Social Networking Services (SNS) and Big Data, the exponential growth of user-generated content has resulted in a vast amount of garbage data, such as spam, irrelevant information, fake accounts, and various forms of content pollution. These undesirable data elements not only create clutter within the SNS platforms but also pose serious threats to user experience and the overall integrity of data-driven operations.

The challenge at hand is multifaceted. First, it involves devising an effective garbage data filtering algorithm that integrates advanced machine learning techniques to distinguish, categorise, and subsequently remove or flag these unwanted data instances. Furthermore, this algorithm must not only operate with high precision but also adapt to the ever-evolving landscape of garbage data in SNS.

The primary objective is to find an effective garbage data filtering solution for Social Networking Services (SNS) that should proficiently identify and eliminate unwanted data, including spam, irrelevant content, and fraudulent accounts, thereby improving the user experience by reducing platform noise and bolstering data privacy while mitigating misinformation risks. Furthermore, its successful implementation can play a pivotal role in content curation, sentiment analysis, and user profiling, all of which are crucial for elevating user engagement and the competitiveness of SNS platforms.

The goal of this problem statement is to find a reliable algorithm that works well at removing junk material from the constantly changing social networking services (SNS) landscape, making the internet a safer and more dependable place.

* **Problem Solution from Previous Work (Existing System):**

Naive Bayes is a simple and probabilistic machine learning algorithm used for classification and text analysis. It is based on Bayes' theorem and is called "naive" because of the simplifying assumption that all features are independent of each other, which rarely holds true in real-world data. Despite this simplification, Naive Bayes often works well and is computationally efficient.

**Key Components:**

* 1. **Bayes' Theorem:** Naive Bayes relies on Bayes' theorem, which calculates the probability of an event based on prior knowledge of conditions that might be related to the event.
  2. **Independence Assumption:** The "naive" part of the algorithm assumes that the features used for classification are conditionally independent, meaning that the presence or absence of one feature does not affect the presence or absence of another.
  3. **Training:** During the training phase, the algorithm learns the conditional probabilities of each feature given a class label. This involves counting the occurrences of each feature in different class instances.
  4. **Classification:** In the testing phase, Naive Bayes calculates the probability of a given set of features belonging to each class. The class with the highest probability is chosen as the predicted class for the input.

**Applications:**

* *“Text Classification”:* Naive Bayes is commonly used for text classification tasks, such as spam detection and sentiment analysis.
* Medical Diagnosis: It has applications in medical diagnosis, where it is used to classify diseases based on symptoms.
* Email Filtering: In email systems, it is used to classify emails as spam or not.

**Advantages:**

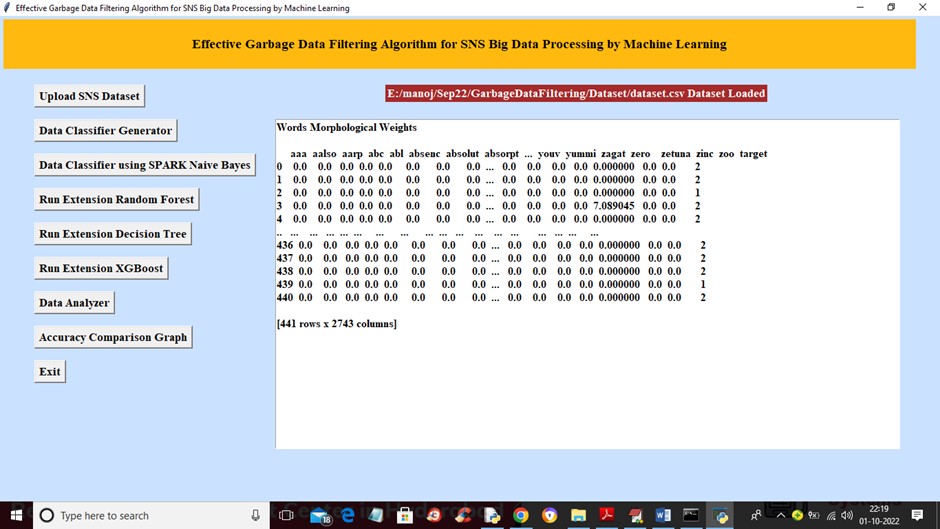
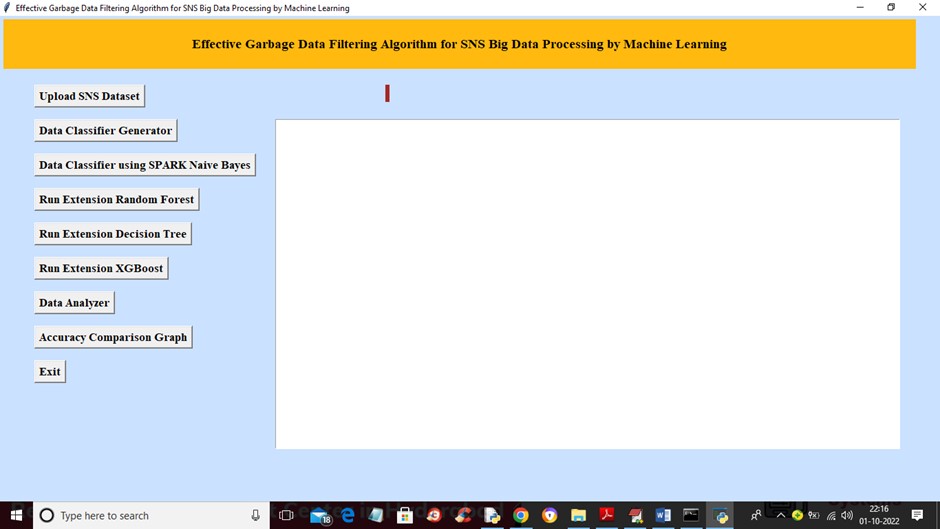
* Simple and easy to implement.
* Works well with high-dimensional data.
* Computationally efficient, making it suitable for real-time classification.

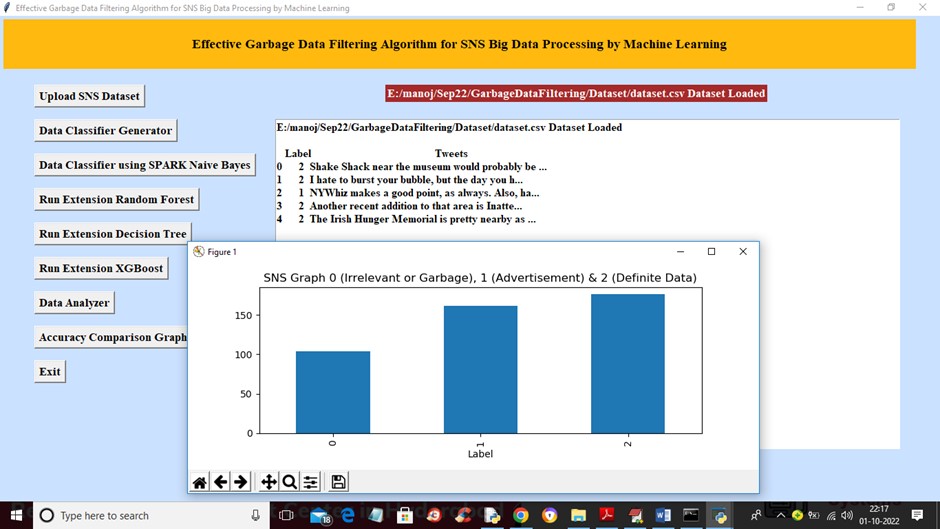
**Limitations:**

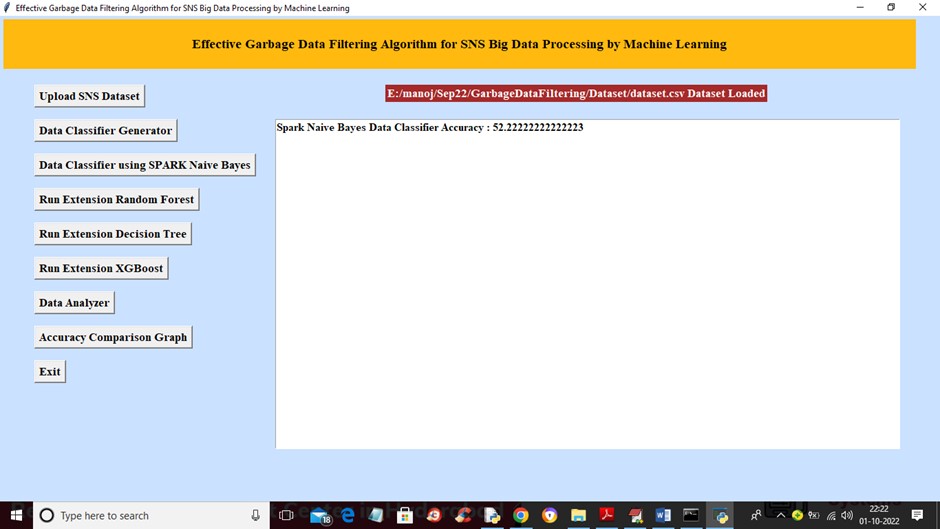
* The independence assumption does not hold in many real-world scenarios.
* Can be sensitive to irrelevant features.
* Requires a relatively large amount of training data for accurate results.

Despite its simplicity and the "naive" assumption, Naive Bayes is a powerful and widely used algorithm, particularly in applications where computational efficiency and simplicity are important.

* **Experiment Result from Previous Work:**







* **Proposed Idea:**

**Decision Tree:**

Using a decision tree over Naive Bayes depends on the specific characteristics of your data and the goals of your classification or data filtering task. Here are some reasons why you might choose to use a decision tree over Naive Bayes:

1. **Handling Complex Relationships:**

Decision trees are capable of modelling complex, non-linear relationships in the data. If your data has intricate interactions between features that Naive Bayes cannot capture due to its independence assumption, a decision tree might be a better choice.

1. **Interpretability:**

Decision trees are highly interpretable. They provide a clear decision-making process that can be easily understood by humans. If you need to explain and justify the classification decisions to stakeholders or auditors, decision trees are more transparent than Naive Bayes.

1. **Mixed Data Types:**

Decision trees can handle both categorical and numerical data seamlessly. If your dataset contains a mix of these data types, a decision tree can be more convenient than Naive Bayes, which typically requires preprocessing for mixed data.

1. **Outliers and Noisy Data:**

Decision trees are relatively robust to outliers and noisy data points. Naive Bayes can be sensitive to such data anomalies, potentially leading to less accurate classifications.

1. **Feature Importance:**

Decision trees can provide insights into feature importance. They can identify which features have the most significant impact on the classification, which can be valuable for feature selection and data understanding.

1. **Scalability:**

Decision trees can handle large datasets and high-dimensional data efficiently. If you have a substantial amount of data, they may be more suitable than Naive Bayes.

1. **Data Imbalances:**

Decision trees can handle imbalanced datasets more effectively. They can be adjusted to give appropriate weight to minority class samples, improving classification performance in such scenarios.

1. **Data Evolution:**

If your dataset is dynamic and changes over time, decision trees can be updated and adapted with new data, ensuring the model remains accurate as data patterns evolve. Naive Bayes is typically a static model once trained.

However, it is essential to note that the choice between Naive Bayes and decision trees should be based on a thorough understanding of your data, the problem you are trying to solve, and the specific requirements of your project. Sometimes, Naive Bayes might perform well, especially when the independence assumption roughly holds, and simplicity and speed are critical. In other cases, decision trees may be a more suitable choice, given their ability to capture complex relationships and provide interpretability.

**Random Forest:**

Using Random Forest over Naive Bayes depends on the characteristics of your data and the specific goals of your classification or data filtering task. Here are some reasons why you might choose to use a Random Forest over Naive Bayes:

1. **Handling Complex Relationships:**

Data connections that are complicated and non-linear can be modelled using random forests. A Random Forest could be a preferable option if your data has complex relationships between attributes that Naive Bayes cannot detect because of its independence premise.

1. **Ensemble Learning:**

An ensemble learning technique called Random Forest uses many decision trees to provide predictions. Comparing this ensemble technique to a single Naive Bayes classifier frequently yields greater accuracy and resilience.

1. **Reducing Overfitting:**

Overfitting, which happens when a model memorises the training data rather than generalising from it, is less likely to happen with Random Forests. In some cases, overfitting might be more of a problem for naive Bayes models.

1. **Handling Noisy Data:**

As they aggregate the predictions of various trees, which have the tendency to cancel out noise, Random Forests are better able to handle noisy data.

1. **Feature Importance:**

With the use of Random Forests, you can determine which factors have the greatest impact on your ability to make predictions. For feature selection and data comprehension, this knowledge may be useful.

1. **Outliers and Imbalanced Data:**

Compared to Naive Bayes, Random Forests are better able to manage outliers and unbalanced datasets. They may be modified to give minority class samples the proper weight, enhancing classification performance in such circumstances.

1. **Adaptability to New Data:**

If your dataset is dynamic and changes over time, Random Forests can be updated and adapted with new data, ensuring the model remains accurate as data patterns evolve. Naive Bayes is typically a static model once trained.

1. **Robustness to Irrelevant Features:**

Random Forests may naturally overlook unimportant traits throughout the tree-building process; therefore, they are less impacted by them.

1. **Scalability:**

Random Forests can handle large datasets and high-dimensional data efficiently.

1. **Model Averaging:**

A method of model averaging offered by Random Forests lowers the possibility of a single tree delivering an unfavourable forecast.

It is important to remember, nevertheless, that your decision about Random Forest vs. Naive Bayes should be based on a comprehensive knowledge of your data, the issue you are attempting to address, and the needs of your project. With well-behaved data, naive Bayes is a viable option for smaller situations and is computationally effective. In general, complicated, noisy, or huge datasets are better suited for using Random Forests because of their resilience and ensemble nature.

**XGBoost**

The decision to use XGBoost versus Naive Bayes is influenced by the specific properties of your data as well as the aims of your classification or data filtering operation.

Here are some advantages of using XGBoost, a gradient boosting technique, over Naive Bayes:

1. **Handling Complex Relationships:**

XGBoost is a powerful algorithm for modelling complex, non-linear relationships in the data. It can capture intricate interactions between features that Naive Bayes, with its independence assumption, cannot handle effectively.

1. **Ensemble Learning:**

XGBoost is an ensemble learning method that combines multiple decision trees to make predictions. It often results in improved accuracy and robustness compared to a single Naive Bayes classifier.

1. **High Accuracy:**

XGBoost is known for its high predictive accuracy. It is particularly effective in scenarios where you need the highest possible accuracy, such as in predictive modelling or competitions.

1. **Feature Importance:**

XGBoost can provide insights into feature importance, helping you identify which features are most influential in making predictions. This information can be valuable for feature selection and data understanding.

1. **Robustness to Irrelevant Features:**

XGBoost is less affected by irrelevant features, as it can naturally ignore them during the tree-building process.

1. **Model Tuning:**

XGBoost offers a wide range of hyperparameters that can be fine-tuned to optimise model performance, making it a versatile choice for different datasets and problems.

1. **Handling Noisy Data:**

XGBoost can handle noisy data more effectively, as it aggregates the predictions of multiple trees, which tend to cancel out noise.

1. **Scalability:**

XGBoost can handle large datasets and high-dimensional data efficiently.

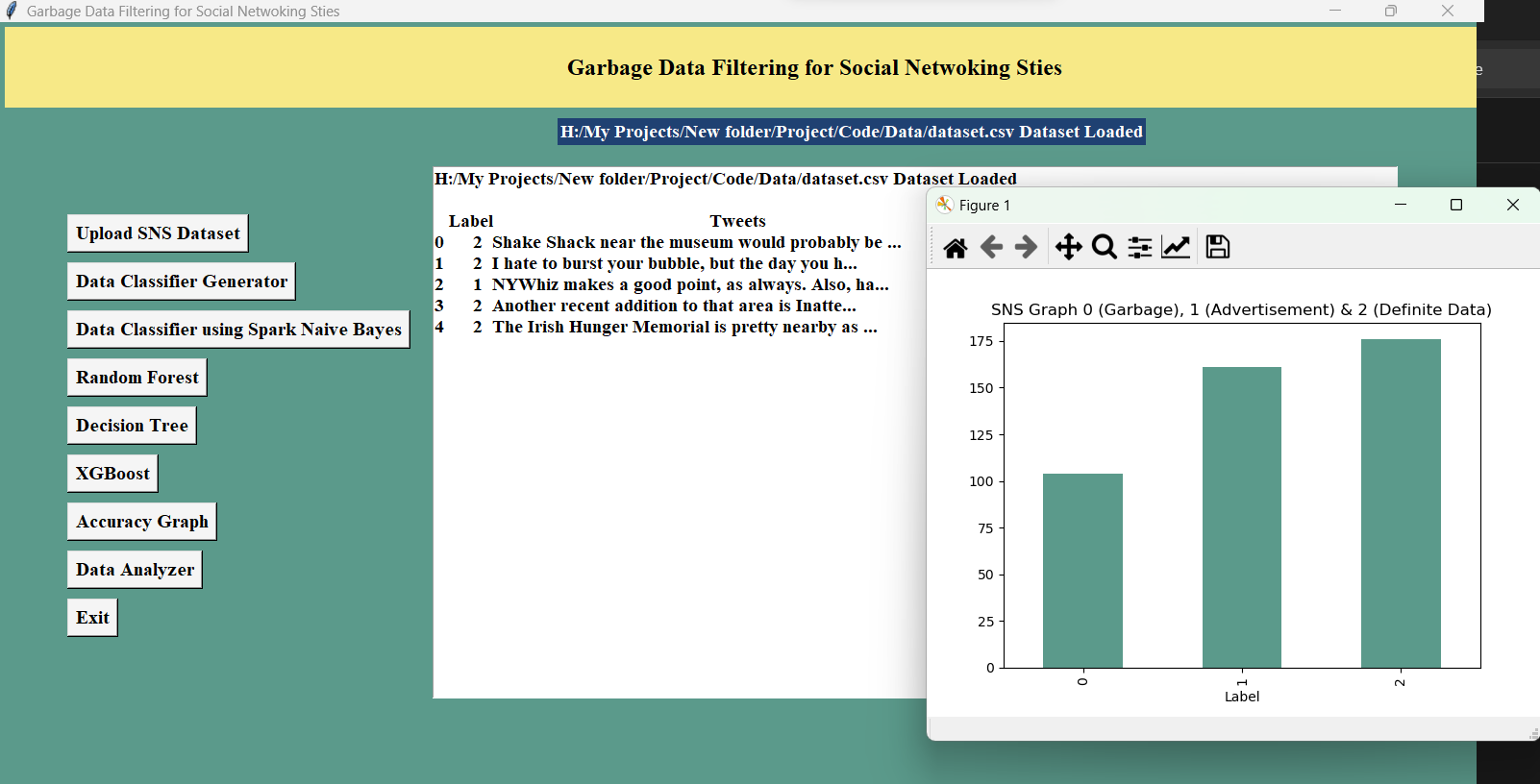
1. **Handling Imbalanced Data:**

XGBoost can be adapted to handle imbalanced datasets more effectively than Naive Bayes, by adjusting class weights or using specialised techniques.

1. **Advanced Techniques:**

XGBoost incorporates advanced techniques, such as gradient boosting and regularised learning, which make it a strong choice for many machine learning tasks.

However, it is essential to note that the choice between XGBoost and Naive Bayes should be based on a thorough understanding of your data, the problem you are trying to solve, and the specific requirements of your project. XGBoost is typically more suitable for complex, high-dimensional, or challenging datasets, where its ensemble nature and robustness to outliers and noise provide significant advantages. Naive Bayes, on the other hand, is a simpler, probabilistic method that can be useful in specific scenarios, particularly when the independence assumption roughly holds, and computational efficiency is critical.

**Comparison between Proposed Algorithm and Existing Ones**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated**

**References:**

1. Lei Yu, Guangyao Pan, Mengning, “Garbage Detection Algorithm Based on Deep Learning,” ITNEC. vol. 5, pp. 1-14, 2021.
2. II-Kyu Ha, Bong-Hyun Back, “Effective Garbage Data Filtering Algorithm for SNS Big Data Processing by Machine Learning,” ICAIIC. vol. 2020, pp. 01-05, 2020.
3. Xinyi Sun, Liming Du, Yanshuang Wang “Text Classification in Architecture Field Based on Naive Bayes Algorithm,” ICDIIME. vol. 2022, pp. 1-6, 2022.
4. Chingmuankim, Rajni Jindal, “Classification and Analysis of Textual data using Naive Bayes with TF-IDF,” ICECIE. vol. 4, pp. 01-07, 2023.