VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Machhe, Belagavi, Karnataka 590018

INTERNSHIP REPORT

on

"SENTIMENT ANALYSIS USING NLP"

Submitted in partial fulfillment of the requirement
for the award of the degree of

Bachelor of Engineering
in
Information Science and Engineering
by

Umaid Manzoor[1BG21IS060]

Vidyayāmruthamashnuthe

B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

Department of Information Science and Engineering

2023 – 2024

B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

Vidyayāmruthamashnuthe

CERTIFICATE

Certified that **Umaid Manzoor(1BG21IS060)** has carried out the Internship on **SENTIMENT ANALYSIS USING NLP** conducted from **4th October to November 3rd 2023.** Candidate has fulfilled all the requirements prescribed in the curriculum. Candidate has incorporated all the corrections/suggestions indicated during the Internal Assessment. The internal internship work has been approved as it satisfies the curriculum requirements.

Ms. Yashaswini B V	**Dr. S Srividhya**	**Dr. Krishnamurthy G N**
Assistant Professor,	**Professor & Head,**	**Principal,**
Dept. of ISE, BNMIT	**Dept. of ISE, BNMIT**	**BNMIT**

B.N.M. Institute of Technology

An Autonomous Institution under VTU, Approved by AICTE

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

Vidyayāmruthamashnuthe

DECLARATION

I, **Umaid Manzoor(1BG21IS060)** a student of B.E. in Information Science and Engineering, Bengaluru, hereby declare that, I have carried out one month internship at **BNM Institute of Technology, Post Box No.7087, 27th Cross, 12th Main, Banashankari 2nd Stage, Bengaluru-560070** from **4th October** to **3rd November 2023** under the guidance of **Ms. Yashaswini B V**. This internship report is submitted to Visvesvaraya Technological University, Belagavi in partial fulfillment of the requirement for the award of degree of Bachelor ofEngineering in Information Science & Engineering during the academic year 2023-24.

Place: Bengaluru **Signature of Student**

Date:

ACKNOWLEDGEMENT

I consider it a privilege to express through the pages of this report, a few words of gratitude to all those distinguished personalities who guided and inspired me in the completion of this internship.

I would like to thank **Shri. Narayan Rao R Maanay**, Chairman, BNMEI, Bengaluru for providing an excellent academic environment in college.

I would like to thank **Prof. T. J. Rama Murthy**, Director, BNMIT, Bengaluru for having extended his support and encouragement during the course of work.

I would like to thank **Dr. S. Y. Kulkarni**, Additional Director, BNMIT, Bengaluru for his extended support and encouragement during the course of work.

I would like to express my gratitude to **Prof. Eishwar N Maanay**, Dean, BNMIT, Bengaluru for his relentless support, guidance, and encouragement.

I would like to thank **Dr. Krishnamurthy G.N.**, Principal, BNMIT, Bengaluru for his constant encouragement.

I would like to thank **Dr. S. Srividhya**, Professor and Head of the Department of Information Science and Engineering, BNMIT, Bengaluru, for her support and encouragement towards the completion of the internship.

I would like to express our gratitude to our guide **Ms. Yashaswini B V,** Assistant Professor, Department of Information Science and Engineering, BNMIT, Bengaluru, who has given us all the support and guidance in completing the internship successfully.

I would like to thank Internship coordinator**, Mrs. Divyashree S R,** Assistant Professor**,** Department of Information Science and Engineering, BNMIT, for being the guiding forcetowards the successful completion of the internship.

Umaid Manzoor
1BG21IS060

Table of Contents

List of Figures

CHAPTER 1

INTRODUCTION

BNM Institute of Technology successfully conducted an immersive internship program on SENTIMENT ANALYSIS USING NLP, in collaboration with Pantech E-Learning from 4th October, 2023 to 3rd November, 2023. The program provided students with hands-on experience in cutting-edge security technologies, equipping them with practical skills to navigate the dynamic landscape of cybersecurity.

1.1 Sentiment Analysis using NLP

In an era dominated by vast amounts of textual data, understanding the sentiments expressed within written content has become paramount. Sentiment analysis, a branch of Natural Language Processing (NLP), offers a powerful solution to decipher the emotional undertones of text. This project delves into the realm of sentiment analysis, employing advanced NLP techniques to unravel the sentiments encapsulated in diverse textual data.

Sentiment analysis, also known as opinion mining, involves the use of computational methods to determine and categorize the emotional tone behind a piece of text. Whether it be social media comments, product reviews, or news articles, the ability to gauge sentiments provides invaluable insights for businesses, researchers, and decision-makers alike.

This project aims to contribute to the evolving landscape of sentiment analysis by harnessing the capabilities of NLP algorithms. By leveraging machine learning models, we seek to develop a robust system that can not only identify positive, negative, or neutral sentiments but also discern the nuances and intricacies of human expression within the text.

As we embark on this journey, the overarching goal is to enhance our understanding of the sentiment dynamics present in textual data. Through a thoughtful integration of NLP methodologies, our project strives to make sense of the vast sea of unstructured text, enabling us to extract meaningful insights and make informed decisions.

CHAPTER 2

INTERNSHIP OBJECTIVES

The internship aims to provide participants with a comprehensive skill set to proficiently apply Natural Language Processing (NLP) techniques for sentiment analysis. Participants will cultivate ethical practices, critical thinking, and collaborative problem-solving abilities in the context of analyzing sentiments within textual data. The following are the primary objectives of the internship:

1. Understanding Sentiment Analysis Fundamentals
2. Distinguishing NLP, Machine Learning, and Deep Learning in Sentiment Analysis
3. Exploring Real-World Applications of NLP in Sentiment Analysis
4. Addressing Ethical and Legal Considerations in NLP and Sentiment Analysis
5. Integrating NLP Technologies into Sentiment Analysis Solutions
6. Cultivating Critical Thinking and Problem-Solving Skills in Sentiment Analysis
7. Effectively Communicating NLP Concepts in Sentiment Analysis
8. Staying Informed on Industry Trends in NLP and Sentiment Analysis
9. Formulating a Holistic Understanding of the Relationship Between NLP and Sentiment Analysis
10. Recognizing Broader Implications, Challenges, and Opportunities in Leveraging NLP for Sentiment Analysis

CHAPTER 3

SYSTEM REQUIREMENTS SPECIFICATION

System Requirements Specification delineates the specifications that users' devices must meet to successfully engage with the program content. This includes delineating the required software, such as the designated integrated development environment (IDE) and relevant programming languages, as well as the hardware prerequisites, such as processor speed, memory, and network capabilities.

3.1 User Requirements

Learning Objectives Alignment: Users seek program content that aligns with their specific goals and interests in AI and cybersecurity.

Interactivity and Engagement: Preferences lean towards interactive learning methods, practical exercises, and hands-on projects for enhanced engagement.

Flexibility and Accessibility: Users value flexible program formats, allowing asynchronous access and accommodating diverse schedules.

Depth of Coverage: Expectations vary, with some users desiring a balanced approach suitable for both beginners and those with advanced knowledge.

Practical Application Opportunities: Users express a need for real-world application scenarios, case studies, or projects to apply AI concepts practically.

Support and Communication: Expectations include accessible support mechanisms, such as discussion forums, mentorship, and clear communication channels.

Ethical and Legal Dimensions: Given the sensitivity of AI in cybersecurity, users emphasize comprehensive coverage of ethical considerations and legal frameworks.

3.2 Software Requirements

Operating System: Windows 7 or later.

Python: Widely used for coding, data preprocessing, model training, and evaluation.

Jupyter Notebook: Used to create and share documents that contain live code, equations, visualizations, and narrative text.

Scikit-learn: A widely used Machine Learning (ML) library.

NumPy: Used for performing linear algebraic operations.

Pandas: Especially used for manipulating data in files of different formats.

PyPlot (Matplotlib): Used for plotting and obtaining visualizations.

Seaborn: Improves the look-and-feel of a visualization.

3.3 Hardware Requirements

Computer with Adequate Processing Power: Ensure the system has a processor suitable for running resource-intensive AI algorithms and simulations. Multi-core processors (e.g., Intel Core i5 or equivalent) are preferred.

Sufficient RAM: A minimum of 8GB RAM, although 16GB or more is preferable for smooth handling of AI and cybersecurity applications.

Graphics Processing Unit (GPU): If possible, use a system with a dedicated GPU, as many AI frameworks leverage GPU acceleration for faster computations. At least 1GB

Storage Space: Have ample storage space, as datasets in AI and cybersecurity can be substantial. SSDs are recommended for faster data access. SSD with at least 256GB is recommended.

Virtualization Support: Ensure hardware virtualization support in the CPU for optimal performance in virtualized environments.

CHAPTER 4

METHODOLOGY

The methodology section outlines the systematic approach undertaken to conduct sentiment analysis using Natural Language Processing (NLP) techniques. Drawing inspiration from cybersecurity initiatives, this research design merges the power of artificial intelligence with linguistic analysis to discern and interpret sentiments within textual data.

4.1 Defining the Problem

This section outlines the systematic process employed to clearly define the problem of sentiment analysis using Natural Language Processing (NLP) techniques. Drawing inspiration from the approach taken in malware detection, we aim to articulate the objectives, scope, and distinguishing characteristics of sentiments within the textual dataset.

4.2 Techniques Used

Data Collection: Gather a diverse dataset comprising various types of textual data. Include samples with different sentiment polarities (positive, negative) and ensure representation from multiple sources. Document the sources and any preprocessing steps, such as text normalization or removal of irrelevant information.

Exploratory Data Analysis (EDA): Conduct an extensive exploratory analysis of the textual dataset. Understand the distribution of sentiment categories, identify patterns, and visualize key features. EDA informs subsequent steps, aiding in feature selection and providing insights into the nature of sentiments in the dataset.

Data Pre-processing: Clean and preprocess the textual dataset by addressing issues like missing values, standardizing text, and handling imbalances between different sentiment categories. Implement techniques such as oversampling or under sampling to ensure a balanced dataset for effective model training.

Feature Engineering: Identify relevant linguistic features for sentiment analysis and consider engineering new features if necessary. This step enhances the model's capability to capture nuanced patterns associated with different sentiments.

Model Selection: Choose appropriate NLP techniques and machine learning algorithms for sentiment analysis. Explore methods like Natural Language Understanding (NLU) and sentiment lexicon-based approaches. Consider algorithms such as Recurrent Neural Networks (RNNs) or Transformer models for comparison.

Train Test Split: Divide the dataset into training and testing sets, maintaining a reasonable ratio. This ensures the model's ability to generalize to new, unseen data.

Model Training: Train the selected machine learning models using the training dataset. Fine-tune hyperparameters, if necessary, to optimize model performance.

Model Evaluation: Assess the models' performance using the test dataset. Calculate accuracy scores, precision, recall, and F1 score. Generate confusion matrices to visualize true positives, true negatives, false positives, and false negatives for each algorithm.

Model Comparison: Compare the performance metrics of different algorithms. Highlight why Random Forest outperformed others, discussing its strengths and potential limitations in the context of malware detection.

Robustness Testing: Assess the robustness of the chosen model(s) by introducing new, previously unseen data. Evaluate how well the model generalizes to real-world scenarios.

Ethical Considerations: Discuss any ethical considerations related to the use of sentiment analysis models, such as privacy concerns or potential biases in the textual dataset.

Documentation and Reproducibility: Provide comprehensive documentation of the entire sentiment analysis methodology, including code, data sources, and preprocessing steps. Ensuring reproducibility enhances transparency and facilitates future research or validation.

CHAPTER 5

IMPLEMENTATION

In the implementation phase of the internship project, the outlined methodology is translated into action, leveraging a carefully chosen set of tools and techniques. The implementation section provides a granular view of our hands-on approach. The software employed, steps taken for feature engineering, model development, evaluation, and ethical considerations are discussed, offering a comprehensive narrative of the practical execution of our cybersecurity solution.

5.1 List of Modules

Pandas: For efficient data manipulation and analysis.

NumPy: Essential for numerical operations and array manipulations.

Scikit-learn: A comprehensive machine learning library for model development, training, and evaluation.

Matplotlib and Seaborn: For creating visualizations to interpret results.

5.2 Module Description

5.2.1 Pandas Module

Pandas is a powerful and widely used open-source data analysis and manipulation library for Python. It provides easy-to-use data structures and functions designed to make working with structured data seamless. Pandas is particularly well-suited for tasks such as cleaning, transforming, analyzing, and visualizing data.

DataFrame and Series: The core data structures in Pandas are the DataFrame and Series. A DataFrame is a two-dimensional, labeled data structure resembling a table, while a Series is a one-dimensional array-like structure.

Data Manipulation: Pandas simplifies data manipulation with functions for merging, reshaping, and aggregating datasets. It supports SQL-like operations, making it convenient for data manipulation tasks.

Missing Data Handling: Pandas provides tools for handling missing data, allowing users to fill, drop, or interpolate missing values easily. This is crucial formaintaining data integrity during analysis.

Time Series and Date Functionality: The library excels in handling time series data, offering date and time manipulation functionalities. This is particularly valuable for analyzing temporal patterns in datasets.

Data Alignment and Indexing: Pandas aligns data automatically based on labels, simplifying tasks like merging datasets on common columns. The flexible indexing system allows for efficient data retrieval and manipulation.

Input/Output Tools: Pandas supports various file formats, including CSV, Excel, SQL, and more. This facilitates seamless data import and export, making it easy to work with data from different sources.

Statistical Analysis: The library includes statistical functions for descriptive statistics, correlation analysis, and other common statistical measures. This aids in gaining insights into the distribution and characteristics of the data.

Data Visualization: While not a visualization library itself, Pandas integrates well with visualization libraries like Matplotlib and Seaborn. It allows for quick and convenient data plotting and exploration.

5.2.2 NumPy Module

NumPy is a fundamental open-source library for numerical computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays. NumPy is a cornerstone library in the Python scientific computing ecosystem and is widely used in fields such as

N-dimensional Arrays: NumPy's core feature is the numpy.ndarray, a multi-dimensional array object that can represent vectors, matrices, and higher-dimensional data structures. This allows for efficient storage and manipulation of numerical data.

Vectorized Operations: NumPy enables vectorized operations, where mathematical operations are performed on entire arrays, eliminating the need for explicit looping. This results in faster and more concise code.

Broadcasting: Broadcasting is a powerful feature that allows NumPy to perform operations on arrays of different shapes and sizes. This makes it possible to work with arrays that might not be explicitly compatible.

Mathematical Functions: NumPy provides a rich set of mathematical functions for operations such as trigonometry, logarithms, exponentiation, and more. These functions operate element-wise on arrays, making them versatile for various applications.

Linear Algebra Operations: NumPy includes a comprehensive set of linear algebra functions, such as matrix multiplication, decomposition, eigenvalue calculations, and more. This makes it a valuable tool for tasks involving linear algebra. It provides a rich set of mathematical functions for different kind of operations such as exponentiation, logarithms, etc.

Random Number Generation: NumPy has a sub-module (numpy.random) dedicated to generating random numbers. This is particularly useful for applications like simulations and statistical sampling.

Integration with Other Libraries: NumPy seamlessly integrates with other Python libraries, including data visualization libraries like Matplotlib and data manipulation libraries like Pandas. This makes it a foundational component in the Python scientific stack.

5.2.3 Scikit-learn Module

The 'scikit-learn' module, often referred to as sklearn, is an open-source machine learning library for Python. It provides simple and efficient tools for data mining and data analysis, built on top of other popular scientific computing libraries like NumPy, SciPy, and Matplotlib.

Consistent API: scikit-learn maintains a consistent and easy-to-use API across various machine learning algorithms. This consistency simplifies the process of switching between different models and algorithms.

Wide Range of Algorithms: The library includes a broad selection of machine learning algorithms for classification, regression, clustering, dimensionality reduction, and more. This facilitates experimentation and model selection based on the nature of the data and the problem at hand.

Data Preprocessing: scikit-learn provides tools for data preprocessing, including methods for scaling, normalization, encoding categorical variables, and handling missing values. These preprocessing steps are crucial for preparing data for machine learning models.

Model Evaluation: The library offers a suite of functions for model evaluation, including metrics for accuracy, precision, recall, F1 score, and area under the ROC curve. Cross-validation techniques are also available to assess model performance more robustly.

Feature Selection and Extraction: scikit-learn provides methods for feature selection and extraction, allowing users to choose the most relevant features or transform data into a lower-dimensional representation.

Hyperparameter Tuning: Tools for hyperparameter tuning, such as Grid Search and Randomized Search, assist in finding the optimal set of hyperparameters for a machine learning model. They are also available to assess model performance more robustly.

Ensemble Methods: Ensemble methods like Random Forests and Gradient Boosting are implemented, offering powerful tools for improving model performance and handling complex relationships in data.

Integration with NumPy and Pandas: scikit-learn seamlessly integrates with NumPy arrays and Pandas DataFrames, making it easy to incorporate machine learning into existing data analysis workflows.

5.2.4 Matplotlib/Seaborn Module

Matplotlib and Seaborn are two powerful data visualization libraries in Python that enable the creation of a wide range of static, animated, and interactive visualizations. Matplotlib serves as the foundational library for creating plots, while Seaborn provides a high-level interface for creating aesthetically pleasing statistical graphics.

Customizable Plots: Matplotlib allows users to create highly customizable plots, including line plots, scatter plots, bar plots, histograms, and more. Users have fine-grained control over plot elements such as colors, markers, and annotations.

Publication-Quality Graphics: Matplotlib is designed to produce publication-quality graphics suitable for academic papers, presentations, and reports. Users can save plots in various formats, ensuring compatibility with different document types.

Seaborn Styling: Seaborn builds on Matplotlib and provides a high-level interface for creating aesthetically pleasing statistical visualizations. It comes with built-in themes and color palettes to enhance the visual appeal of plots.

Statistical Visualization: Seaborn simplifies the creation of statistical plots such as box plots, violin plots, and heatmaps. These plots are particularly useful for understanding the distribution and relationships within complex datasets.

Integration with Pandas: Both Matplotlib and Seaborn seamlessly integrate with Pandas DataFrames, allowing for easy visualization of data stored in tabular formats. This integration simplifies the process of creating plots directly from DataFrame objects.

Interactive Plots: Matplotlib supports interactive features for zooming, panning, and adding widgets to plots. This is especially valuable for exploring large datasets or creating interactive visualizations for web applications.

5.3 Algorithm

The Random Forest algorithm is a versatile and powerful machine learning ensemble method that excels in both classification and regression tasks. It belongs to the family of decision tree-based algorithms and is renowned for its robustness, scalability, and ability to handle complex relationships within datasets.

Developed based on the concept of ensemble learning, Random Forest constructs multiple decision trees during training and merges their outputs to achieve a more accurate and stable prediction.

The Random Forest algorithm possesses several attributes that contribute to its widespread popularity and effectiveness, often making it more useful than other algorithms in various scenarios.

Robustness to Overfitting: Random Forest is less prone to overfitting compared to individual decision trees, especially when dealing with noisy or complex datasets. The ensemble nature of Random Forest, combining multiple trees, helps mitigate overfittingby reducing variance.

High Predictive Accuracy: Random Forest tends to achieve high predictive accuracy across a diverse range of datasets. By aggregating predictions from multiple trees, it often produces more reliable results compared to single decision tree models, which can be sensitive to variations in the data.

Handling Nonlinear Relationships: The ensemble of decision trees in Random Forest excels at capturing complex, nonlinear relationships within the data. This makes it suitable for tasks where the underlying patterns may be intricate or difficult to model with simpler algorithms.

Feature Importance Assessment: Random Forest provides a measure of feature importance, indicating the contribution of each feature to the overall predictive performance. This information is valuable for feature selection and gaining insights into the factors driving the model's decisions.

Automatic Feature Selection: The algorithm incorporates implicit feature selection by considering only a random subset of features at each split during the construction of decision trees. This helps in reducing the impact of irrelevant or redundant features, contributing to model efficiency.

Versatility Across Tasks: Random Forest is versatile and can be applied to both classification and regression tasks. Its adaptability to different types of problems makes it a one-stop solution for various machine learning applications.

Out-of-Bag (OOB) Evaluation: The out-of-bag evaluation feature allows for unbiased performance assessment without the need for a separate validation set. This is particularly useful when labeled data is limited, as it maximizes the utilization of available information. This helps in reducing the impact of irrelevant or redundant features, contributing to model efficiency.

Reduced Sensitivity to Hyperparameters: Random Forest is generally less sensitive to hyperparameter tuning compared to certain other algorithms. While tuning is still beneficial, Random Forest often delivers reasonable performance with default parameter settings.

Effective Handling of Missing Data: Random Forest can effectively handledatasets with missing values without requiring imputation. The algorithm leverages the available information in the data, making it robust in scenarios where missing data is common.

5.4 Pre-processing of twitter data

Data may be in unstructured format that is not good for extracting feature. Tweets may consist of empty spaces, stop words, slangs, special characters, hashtag, emoticons, time stamps, abbreviations, URL's etc. for mining these data we should have to pre-process the data first by the using the functions of NLTK. While doing pre-processing our first aim is to extract message then we will remove all hashtags(#), empty spaces, repeating words, stop words(such as he, she, them, the etc.). emoticons and abbreviation will be replaced by their corresponding meaning such as :-), =D, LOL. They will be replaced by happy, laugh and laughing out loudly respectively. After done this thing we are ready to give this pre-process data to our new classifier for further process so we could get our required result.

We did code in python where we define function which would be used to get processed data.

- Remove quotes: give the access to user to eliminate the quotes from the tweet.
- Remove @- give the option to remove the @ symbol, delete @ together with the username or replace @ and the username with a word 'AT_USER' and append to the stop words.
- Remove URL's- URL stands for uniform resource locator. offers options to remove URLs or replace them with the word 'URL' and append to stop words.
- Removal of RT(Re-Tweet)- it deleted the RT word from the text.

- Removal of emoticons- replace emoticons with their correct meaning
- Removal of duplicates- delete all the duplicate word from the tweet.
- Removal of hashtag(#)- remove hashtags from the tweet.
- Removal of stopwords- delete all the stopwords from the tweet such as he, she, them because they do not convey meaning in classification.

CHAPTER 6

RESULTS AND DISCUSSIONS

This chapter consist of various opinion mining result that we have achieved in the implementation.

6.1 Snapshots of the project and description

1. Tweets collected from twitter

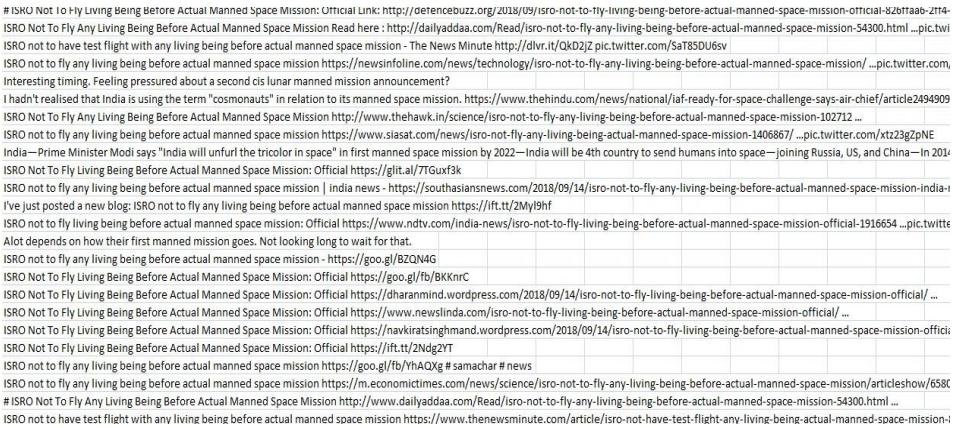

Fig. 6.1 Tweets collected from twitter using Twitter API

So above are the tweets dataset which we have fetch from the twitter with the help twitter API. These twitter dataset contain tweet which are related to “Gaganyaan”. Gaganyaan is a mission of space satellite which is run by Indian government

6.2 Performance metrics of sentiment classification

Generally, to measure the performance of sentiment classification we use some predefined standards such as accuracy, precision, recall. Accuracy is dependent on two measure.

6.2.1 Precision

To get the right value of precision, we divide the total number of rightly classified positive observation by the total number of predicted positive observation. High precision denotes that the observation classified positive is indeed positive.

$$\text{Precision} = \frac{tp}{tp + fp}$$

6.2.2 Recall

It is the ratio between the right classified positive observation to the total number of positive observation.high recall denotes that the class is rightly classified.

$$\text{Recall} = \frac{tp}{tp + fn}$$

6.2.3 Accuracy

In order to find the which model gives better result, then it is necessary to find the accuracy. Accuracy for any model can be given as:

$$\text{Accuracy} = \frac{tp + tn}{tp + fn + tn + fp}$$

6.3 Results of classifier for twitter data

Here, different classifier have tested on same dataset in which some give best performance in terms of precision, recall and accuracy. These are data which we have fetched from the twitter dataset. Here below are performance of some classifier.

- Naïve bayes(NB):

```
NB accuracy:  0.6341463414634146
NB Precision:  0.6271929824561403
NB Recall:  0.89375
```

Fig 6.3.1 Naïve Bayes

Naïve bayes classifer is tested on our dataset. Generally it works on large datasetand it is fast. It the accuracy 63% and precision is 62% and it gives better performance in term of recall i.e, 89%.

- Support vector machine(SVM):

```
SVM accuracy:  0.7896341463414634
SVM Precision:  0.8531468531468531
SVM Recall:  0.7625
```

Fig 6.3.2 Support vector machine (SVM)

Support vector machine classifier generally it works better in small dataset. It give accuracy of 78% and moving on to side of precision and recall 85% and 76% respectively.

- K-nearest neighbor(k-NN):

```
KNN accuracy:  0.7073170731707317
KNN Precission:  0.8557692307692307
KNN Recall:  0.55625
```

Fig 6.3.3 K-nearest neighbor(k-NN)

It is works well in accuracy it gives result 70% precision and recall 85% and 55% respectively.

we collected a total of 1431 tweets from twitter and did sentiment analysis on those tweets by using some supervised learning classifier such as support vector machine, k-nearest neighbor, naïve bayes, decision tree. with the help of these classifier we are able to find standard measure such as accuracy, precision and recall. Performance results of these classifier are mention in table below.

Supervised learning techniques	Accuracy	Precision	Recall
SVM	78.9	85.3	76.25
Naïve Bayes(NB)	63.4	62.7	89.3
Decision tree (DT)	79.5	83.0	82.5
k-NN	70.7	85.5	55.6

Fig 6.3.4 Result of classifier on our dataset

To find the superior one techniques among the all techniques, we have done a comparative analysis. It is found that DT has 79.5% accuracy which is the highest among all whereas naïve bayes has 63.4% accuracy that is least accuracy among all. So we can conclude that DT is best to find the accurate result. Moving on to the precision K-NN got 85.5 precision that is the highest among all. That means k-nn gives substantially relevant results than the irrelevant results. And on the other hand Naïve bayes got 89.3% recall which is the highest among all classifier that means our classifier returned most of the relevant results.

Fig 6.3.5 performance of classifiers

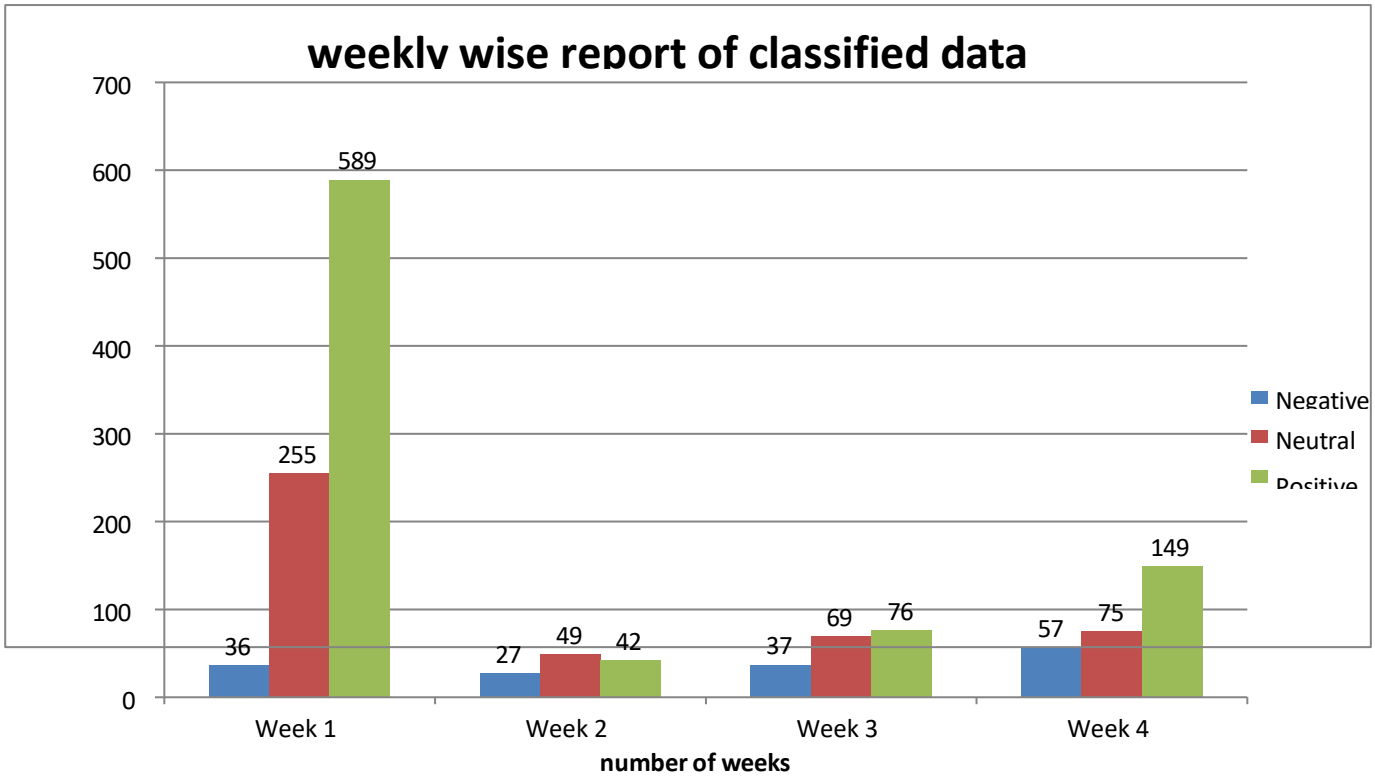

Figure 6.3.6 weekly wise report of classified data

This is the graph between the number of tweets and number of weeks. Which indicates that in each and every week, people has done how many tweets. Such as number of positive tweet and number of negative tweet, number of neutral tweet.

CHAPTER 7

CONCLUSION

In conclusion,in our contemporary data-driven landscape, sentiment analysis emerges as a critical tool for unraveling the complexities of unstructured data from diverse sources. Our investigation into sentiment classifiers highlighted the decision tree as a standout performer, showcasing superior accuracy, precision, and recall, especially in the context of Twitter datasets. By delving into public sentiments expressed through tweets, our project provided valuable insights into societal perspectives.

Looking forward, the potential for advancements is vast. Envisaging a user-friendly web application, we aim to democratize sentiment analysis. Enhancements in handling nuanced sentences and the incorporation of image processing for multimedia sentiment analysis stand as future milestones. Beyond uncovering sentiments, our project lays the foundation for ongoing developments, contributing to the continuous evolution of sentiment analysis methodologies. As organizations increasingly leverage sentiment insights for informed decision-making, our work not only addresses current needs but also anticipates and prepares for the expanding horizons of sentiment analysis in the digital era.

REFERENCES

1. https://arxiv.org/abs/1904.0727
2. https://link.springer.com/article/10.1023/A:1010933404324
3. https://www.sciencedirect.com/science/article/pii/S108480451200219X
4. https://arxiv.org/abs/2004.10385
5. https://pandas.pydata.org/pandas-docs/stable/index.html
6. https://scikit-learn.org/stable/documentation.html
7. https://matplotlib.org/stable/contents.html
8. https://seaborn.pydata.org/documentation.html
9. https://www.geeksforgeeks.org/ml-label-encoding-of-datasets-in-python/
10. https://arxiv.org/abs/2012.00911
11. https://www.sciencedirect.com/science/article/pii/S0360835219303103
12. https://arxiv.org/abs/1903.07279
13. https://arxiv.org/abs/1407.7502
14. https://www.sciencedirect.com/science/article/pii/S036083521830588X
15. https://arxiv.org/abs/1609.04027